VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra kybernetiky a biomedicínského inženýrství

Rozvrhovací systém pro krevní centrum
Blood centre scheduling system

2012 Bc. Pavel Sázel

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra kybernetiky a biomedicínského inženýrství

Zadání diplomové práce

Student: **Bc. Pavel Sázel**

Studijní program: N2649 Elektrotechnika

Studijní obor: 3901T009 Biomedicínské inženýrství

Téma: Rozvrhovací systém pro krevní centrum
Blood Centre Scheduling System

Zásady pro vypracování:

Diplomová práce se zabývá vytvořením aplikace implementující algoritmy pro bezpečný přístup, rozvrhování a provoz krevního centra s možností importu a exportu dat. V souhrnu je práce charakterizována těmito body:

1. Analýza problému provozu krevního centra, respektive jeho informačního systému.
2. Návrh a implementace administrativní části.
3. Návrh a implementace rozvrhovací části.
4. Návrh a implementace veřejně přístupné části.
5. Zhodnocení dosažených výsledků.

Seznam doporučené odborné literatury:

[1] HEROUT, P. *Učebnice jazyka Java.* České Budějovice: Kopp, 2002. 392 s. ISBN 978-80-7232-398-2.
[2] EVJEN, B. - HANSELMAN, S.*ASP.NET 3.5 v jazycích C# a Visual Basic.* Praha: Computer Press, 2009. 1600 s. ISBN 978-80-251-2069-9.
[3] STAUGAARD, A. *Information Systems Programming with Java.* 2nd Edition. Prentice Hall, 2003. 816 s. ISBN 978-0131018600.
[4] KALALI, M. *GlassFish Security.* Packt Publishing, 2010. 296 s. ISBN 978-1847199386.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Zdeněk Slanina, Ph.D.**

Datum zadání: 18.11.2011

Datum odevzdání: 04.05.2012

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA 1

doc. Ing. Jiří Koziorek, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne 4. 5. 2012

.....

podpis

Poděkování

Rád bych poděkoval především vedoucímu mé diplomové práce, Ing. Zdeňku Slaninovi, Ph.D. , za vedení mé diplomové práce a pod nímž se pracovalo velmi dobře a nakonec i úspěšně, a paní Ing. Dagmar Valové za konzultace týkající se uvedení do problematiky a potřebami aplikace pro krevní centrum. Dále bych rád poděkoval rodině za oporu a hlavně otci za časté konzultace nad možnostmi realizace databázové části aplikace.

Abstrakt

Cílem této diplomové práce je návrh, realizace a implementace informačního systému, který by měl sloužit pro krevní centra a rozšířit možnost klientů tohoto centra (dárců), co se přihlašování k odběrům týká, a to přes webové rozhraní. Systém by měl umožňovat na jednotlivé dny a hodiny vyhradit určitý počet míst pro přihlášení na daný typ odběru. Toto rozvržení by měl provádět uživatel s administrátorským přístupem.

Abstract

The aim of this thesis is the design, realization and implementation of information system, which should serve for the blood centre and allow to clients of this centre (donors) use the opportunity of logging to blood draw over the Web. The system allows allocating number of spaces to login for specific kind of blood draw for individual days and hours. This layout should perform a user with administrator access.

Seznam použitých symbolů a zkratek

AJAX (Asynchronous JavaScript a XML) - obecné označení pro technologie vývoje interaktivních webových aplikací, které mění obsah svých stránek bez nutnosti znovunačítání. Od klasických webových aplikací poskytují příjemnější prostředí, ale vyžadují použití moderních webových prohlížečů.

API (Application Programming Interface) - označuje rozhraní pro programování aplikací. Jde o sbírku procedur, funkcí či tříd nějaké knihovny, programu nebo jádra operačního systému, které může programátor využívat.

BSD (Berkeley Software Distribution tzv. Berkeley Unix) je odvozenina Unixu distribuovaná Kalifornskou univerzitou v Berkeley.

CGI (Common Gateway Interface) - je protokol pro propojení externích aplikací s webovým serverem.

GNU Nekompletní počítačový svobodný operační systém projektu GNU. Systém je tzv. UNIX-like a neobsahuje žádný originální kód Unixu. Obvykle používá jádro Linux, jelikož jeho oficiální jádro GNU Hurd ještě nebylo dopsáno.

GPL (General Public License) - je nejpopulárnějším a dobře známým příkladem silně copyleftové licence, která vyžaduje, aby byla odvozená díla dostupná pod toutéž licencí.

HTML (HyperText Markup Language) - hypertextový značkovací jazyk.

HTTP (HyperText Transfer Protocol) - protokol pro přenos hypertextu.

J2EE zkratka pro Java 2 Enterprise Edition - přístup, jak provádět design, vývoj, nasazení a provozování vícevrstvých aplikací pomocí jazyka Java formou několika základních komponent.

JDBC (Java DataBase Connectivity) - technologie, která je základem J2EE, a která je určena pro práci s databázemi.

ODBC (Open DataBase Connectivity) - je standardizované softwarové API pro přístup k databázovým systémům. Snahou ODBC je poskytovat přístup nezávislý na programovacím jazyku, operačním systému a databázovém systému.

ODBMS (Object-oriented DataBase Management System) - objektově orientované softwarové vybavení, které zajišťuje práci s databází - tvoří rozhraní mezi aplikačními programy a uloženými daty.

OS Operační systém.

RDBMS (Relational DataBase Management System) - systém řízení relační báze dat.

SGML (Standard Generalized Markup Language) - univerzální značkovací metajazyk umožňující definovat značkovací jazyky jako své vlastní podmnožiny.

SMTP (Simple Mail Transfer Protocol) - je internetový protokol určený pro přenos zpráv elektronické pošty.

SSL (Secure Socket Layer) - je protokol, resp. vrstva vložená mezi vrstvu transportní a aplikační, která poskytuje zabezpečení komunikace šifrováním a autentizací

komunikujících stran.

TLS (Transport Layer Security) - je následovníkem SSL a taktéž patří do skupiny kryptografických protokolů, poskytující možnost zabezpečené komunikace na Internetu pro služby jako WWW, elektronická pošta, internetový fax a další datové přenosy.

URL (Uniform Resource Locator) - je řetězec znaků s definovanou strukturou, který slouží k přesné specifikaci umístění zdrojů informací na Internetu.

W3C (World Wide Web Consorcium) - mezinárodní konsorcium, jehož členové společně s veřejností vyvíjejí webové standardy pro World Wide Web.

WWW (World Wide Web) - označení pro aplikace internetového protokolu HTTP.

Obsah

1 Úvod

Dnes, kdy moderní technika zasahuje snad úplně do všech koutů lidské činnosti a lidského bytí, nemůže být opomenuto ani lékařství. Od vstupu techniky do něj, již pár desítek let uplynulo a její možnosti v tomto důležitém oboru jsou bezmála úchvatné. I tato práce by měla být zefektivněním činnosti jednoho z mnoha lékařských oborů. Obecně by se tato práce měla zabývat rozšířením možností pacientů k přihlášení na odběr krve. Doposud byly k dispozici pouze možnosti osobního objednání, následovalo objednávání telefonické a v době elektronické pošty i objednávání pomocí e-mailu. Avšak všechny tyto možnosti jsou vcelku zdlouhavé. Komunikace mezi sestřičkou a pacientem přes e-mail a domlouvání určitého volného termínu vhodného jak pro krevní centrum, tak i pro pacienta, zabírá drahocenný čas pro oba. Nakonec se došlo k myšlence informačního systému pracujícím přes webové rozhraní, nabízející dárcům náhled pro všechny vypsané možnosti odběru k určitému datu a stav obsazení této jednotky a nakonec i možnost přihlášení k dárcem vyhovujícímu odběru.

Aby má snaha nad touto aplikací nebyla zbytečná, zkoušel jsem se informovat v pár dalších krevních centrech, mimo jiné i v těch, nacházejících se v hlavním městě, prohledával jsem internetové stránky krevních center v zahraničí a sháněl informace, zda mají zavedený systém přihlašování a velmi mě překvapilo, že jsem nalezl pouze pár krevních center, které pracují s obdobným systémem (USA, Japonsko), který by měl být cílem i této práce, a to rezervační systém a však plně přizpůsoben pro potřeby krevních center.

Síť WWW (World Wide Web) se od svého vzniku rapidně mění a mění se mnoha různými způsoby. Na počátku všeho byl téměř neznámý jazyk HTML (Hyper Text Markup Language), který byl používán fyziky v CERNu ke sdílení vědeckých dokumentů. Byla to novinka, která se následně rozšířila mezi ostatní vědecké obory, a její normou byly pouze textové informace a jednoduchost přístupu k informacím byla nejdůležitější částí rovnice.

Dnešní moderní webové sídlo není pouhým webovým serverem. Řeší problémy ukládání dat a dotazování na ně, zpracování uživatelských požadavků a vytvoření dokumentu, který obsahuje příslušné informace.

Tato práce se zabývá vytvořením internetové aplikace pro krevní centrum, která by měla usnadnit proces přihlašování k odběrům jak dárcům, tak i zaměstnancům krevního centra. Doposud byly tyto možnosti pouze tři. Dárce se objednal na další odběr osobně na krevním centru, telefonicky nebo e-mailem. To vše však bylo relativně zdlouhavé a neefektivní pro obě strany. Výhodou internetové aplikace je, že klient nemusí instalovat na svém počítači žádný software. K ovládání mu postačí pouze internetový prohlížeč s aktivními cookies a JavaScriptem. Další výhodou internetové aplikace je snadnější aktualizace a rychlejší servis.

Tato aplikace, mimo jiné, obsahuje dynamicky generovaný kalendář, který umožňuje přihlášení na vybrané vypsané odběrové datum. Zároveň bere v úvahu omezení dárců v přihlašování po již vykonaném odběru. Interval, ve kterém dárce nesmí jít na odběr, je pro každý typ odběru různě dlouhá a legislativou měněná. Proto má uživatel s nejvyššími právy tuto hodnotu nastavovat.

Aplikace je rozdělena na tři oblasti přístupné pro uživatele s různými právy.

První oblastí je část pro dárce. Zde má dárce možnost přihlášení k vypsaným odběrům, odhlášení z již přihlášených odběrů a změnu svých kontaktních údaje a přístupového hesla.

Druhou oblastí je část pro zaměstnance krevního centra s přístupovými právy "Organizátor". V této části aplikace má uživatel možnost vypisování krevních odběrů, jejich následnou editaci či jejich úplné zrušení. Dále má správu nad dárci jako je změna jejich osobních, kontaktních či dárcovských informací. K dárcovským informacím patří i povolené typy odběrů pro daného dárce.

Aplikace do jisté míry může být aktivně využívána i pro dárce, kteří nemají přístup k internetu, a to v zastoupení zaměstnance krevního centra. Ten má možnost přihlásit nebo odhlásit dárce k odběru.

V případě, kdy by bylo potřeba informovat určitého dárce, nebo skupinu dárců, aplikace zahrnuje i možnost zaslání hromadné elektronické pošty.

Poslední částí aplikace pro uživatele s právem "Admin" má přístup ke všem funkcím a možnostem aplikace, jako část organizační. Tento uživatel má správu nad celkovou organizací odběrů. Může aktivně nastavovat časové rozmezí odběrů v jednotlivých dnech v týdnu, průměrný čas potřebný k odběru u jednoho dárce, a také celkový počet pacientů, kteří se mohou v této časové jednotce přihlásit k odběru. Má také právo určit počet dní, po které pacient po určitém odběru nesmí darovat.

2 Vývojové prostředky

2.1 Complex Web Server[1] [13]

Complex Web Server je moderní komplexní internetový server s webovým serverem Apache, podporou PHP 5, MySQL 5.1, správcem phpMyAdmin, se správcem služeb a také možností funkce jako Subversion server pro správce verzí. Je vhodný jako pro testování PHP skriptů, tak i jako ostrý internetový server pro provoz webového serveru.

Samotný server je pečlivě nastavený a testovaný. Hlavní výhodou je jeho jednoduchost, kdy již téměř ihned po instalaci je k dispozici webový server s profesionálními možnostmi. I přes jeho jednoduchost je kvalitní, spolehlivý a bezproblémový.

2.1.1 Apache HTTP Server [14]

Apache HTTP Server je softwarový webový server s otevřeným kódem pro GNU/Linux, BSD, Solaris, Mac OS X, Microsoft Windows a další platformy.

Apache podporuje velké množství funkcí, mnoho z nich je implementováno jako komplikovaný modul rozšiřující jádro. Mohou to být funkce podpory programovacích jazyků na straně serveru nebo různá autentizační schémata. Podporované jazyky jsou například Perl, Python, Tcl nebo PHP. Dalšími funkcemi serveru Apache je podpora SSL, TLS, proxy modul, URL rewriter, konfigurace souborů logu.

Přestože hlavním cílem Apache není jeho výkon, může se i tak srovnávat s ostatními webovými servery. Místo implementování jedné architektury, Apache poskytuje mnoho tzv. MultiProcessing modulů, což mu dovoluje přizpůsobit se potřebám systému, na kterém pracuje.

2.1.2 MySQL [15]

MySQL je databázový systém, vytvořený švédskou firmou MySQL AB. Je považován za úspěšného průkopníka dvojího licencování - je k dispozici pod bezplatnou licencí GPL, tak pod komerční placenou licencí.

MySQL je multiplatformní databáze. Komunikace s ní probíhá pomocí jazyka SQL. Podobně jako u ostatních SQL databází se jedná o dialekt tohoto jazyka s některými rozšířeními.

MySQL bylo od počátku optimalizováno především na rychlost, a to i za cenu některých zjednodušení - jednoduché způsoby zálohování, do nedávna bez podpory pohledů, uložených procedur a triggerů.

2.1.2.1 InnoDb

InnoDb je jeden z několika formátů úložiště dat (storage engine) v databázovém systému MySQL. InnoDb byl navržen pro zpracování transakcí - konkrétně pro zpracování mnoha krátkodobých transakcí, které se málokdy anulují.

Vlastnosti:

- podpora transakcí
  Distribuované XA transakce na straně serveru - schopnost více prostředků vstoupit do a účastnit se jedné transakce
- cizí klíče

[1] Tento software je dostupný jako freeware na adrese http://ponkrac.net/complex-web-server/cs

- uzamykání na úrovni řádku
- body obnovení

V této práci je použito úložiště InnoDb, a to právě kvůli dvěma klíčovým výhodám - funkčnost transakcí a podpora cizích klíčů.

2.2 HTML a XHTML [11]

2.2.1 HTML

HyperText Markup Language, označovaný zkratkou HTML, je značkovací jazyk pro hypertext. Je jedním z jazyků pro vytváření stránek v systému WWW (World Wide Web), který umožňuje publikaci dokumentů na internetu.

Jazyk je aplikací dříve vyvinutého rozsáhlého univerzálního značkovacího jazyka SGML (Standard Generalized Markup Language). V roce 1989 se pro tvorbu dokumentů obvykle používaly jazyky TeX, PostScript a SGML. Zapotřebí bylo ovšem něco jednoduššího a v roce 1990 byl navržen jazyk HTML a protokol pro jeho přenos v počítačové síti - HTTP (HyperText Transfer Protocol).

Jazyk je charakterizován množinou značek (tagů) a jejich atributů definovaných pro danou verzi. Mezi značky se uzavírají části textu dokumentu, a tím se určuje význam, neboli sémantika, obsaženého textu. Názvy jednotlivých značek se uzavírají mezi úhlové závorky "<" a ">". Část dokumentu tvořená otevírací značkou, nějakým obsahem a odpovídající ukončovací značkou tvoří tzv. element (prvek) dokumentu. Atributy jsou pak doplňující informace, které upřesňují vlastnosti elementu. Značky jsou obvykle párové, avšak vyskytují se i nepárové.

2.2.2 XHTML

Definice jazyka HTML skoro vyhovuje pravidlům značkovacího jazyka XML. Přirozenou myšlenkou tedy bylo odstranění několik drobných odlišností a upravit definici tak, aby splňovala nároky jazyka XML. Vznikl tak hypertextový značkovací jazyk XHTML (eXtensible HyperText Markup Language). XHTML ve verzi 1.0 nepřináší oproti HTML žádný nové prostředky, pouze upravuje jeho definici do podoby XML.

2.2.3 Použití HTML

Dle osobního názoru je poměrně jedno, který jazyk je použit. Rozdíly mezi nimi jsou malé a schopnosti prakticky stejné. Stále se najde početná skupina lidí, která tvrdí, že cesta XHTML je slepá. V roce 1999 se konsorcium W3C rozhodlo pokračovat pouze ve vývoji jazyka XHTML a poslední verzí jazyka HTML měla být verze 4.01. Ale v roce 2007 začal, mimo pokračování ve vývoji XHTML, souběžně vývoj HTML verze 5, do jehož specifikace se od roku 2009 dostala i specifikace XHTML 5. Hlavním cílem vývoje této verze je využití současných osvědčených pravidel používaných v XHTML a přidání nových značek omezujících závislost na proprietálních zásuvných modulech. Vzhledem k malým rozdílům bylo použití HTML čistě osobní volbou autora práce.

2.3 PHP [4] [5]

PHP (Hypertext Preprocessor) je skriptovací jazyk, který je zabudovaný na straně serveru. Pro zvolení tohoto skriptovacího jazyku při řešení této diplomové práce oproti jiným

možnostem (ASP, Cold Fusion,…) byla jeho jednoduchost, nejpřirozenější způsob práce s databázemi a nezávislost na platformě, na které by vytvořený software měl být zasazen. Největším kladem však je i to, že se jedná o Open Source[2]. Tyto výhody jsou vcelku důležité, aby tento software měl co nejnižší provozní náklady a zároveň mohl být použit na jakékoliv platformě, kterou uživatel na svém serveru má. Ovšem že existují i další dobré programovací jazyky, které jsou také Open Source, avšak jsou pro webovou aplikaci někdy zbytečně velké. PHP bylo navrženo pro práci na webu a v této oblasti vyniká. Připojování a dotazování databáze je otázkou třech řádků kódu, skriptovací jádro má optimalizovánu dobu odezvy potřebnou pro webové aplikace.

Jazyk PHP se skládá z obvyklých doplňků řídících struktur, druhů proměnných, operátorů, deklarací funkcí a deklarací tříd a objektů. PHP je podporováno na mnoha operačních systémech unixového typu a operačních systémech společnosti Microsoft s podporou prostředí Win32. V unixových systémech je ve většině případů webovým serverem Apache a PHP je většinou používáno jako modul pro něj. PHP tak může být zkompilováno jako modul pro Apache nebo jako samostatný interpret. Apache je také k dispozici pro platformu Windows a i PHP je pro tuto kombinaci podporováno. Distribuce PHP jsou i pro webové servery IIS (Windows NT), PWS (Windows 95/98) či Omni HTTPd. Ovšem jediný server, na kterém PHP může být použito jako modul, je Apache, v ostatních je pouze jako interpret CGI. To, že je použit jako interpret, znamená, že PHP skript je pokaždé interpretován, webový server pro něj vytvoří vždy novou instanci interpretu, který tento skript zpracuje. To má však obvykle za následek snížení výkonu. Jako modul je ovšem ve stejném adresním prostoru jako sám proces webového serveru a poskytuje tedy podstatně vyšší výkon. Při použití PHP jako interpretu nastávají starosti ohledně zabezpečení a nelze v nich použít některé funkce, jako například trvalé spojení s databází.

2.4 JavaScript [10]

JavaScript je základním jazykem vývoje webových aplikací. Ať už na webových stránkách slouží pro jejich interaktivitu nebo vytváří celé aplikace, dnešní web by bez něj nebyl tím, čím je.

Je to na standardech založený jazyk s formální specifikací, avšak každý ze současných prohlížečů tuto specifikaci interpretuje trochu jinak. Pokud jde ale o základní funkce, má v jejich podpoře a interpretaci sbíhavou tendenci.

JavaScript není kompilovaným jazykem, čímž vytváří pocit, že není jazykem dostatečně výkonným. Ze začátku byl odmítán a mnoho webových stránek bylo vytvářeno pomocí hypertextového značkovacího jazyka (HTML) s grafikou, jež postrádala jak vizuální dojem, tak schopnost ovlivňovat obsah stránek. Přijat byl až po objevení jeho užitečnosti a síly, díky níž lze jak simulovat, tak vytvářet interaktivitu na World Wide Webu.

---

[2] Open Source - počítačový software s otevřeným zdrojovým kódem. Otevřenost znamená jak technickou, tak legální dostupnost - licenci, která umožňuje při dodržení jistých podmínek uživatelům zdrojový kód využívat. Toto označení se také používá i pro mnoho vlastností, které se u Open Source programů vyskytují - např. bezplatná dostupnost software.

2.4.1 AJAX (Asynchronous JavaScript And XML)

AJAX je souborem technologií kombinujících JavaScript a přístup k webovému serveru. AJAX se používá pro tvorbu vysoce interaktivních webových aplikací.

Bez použití AJAXu by uživatel webové aplikace musel čekat na odpověď od webového serveru. V aplikaci využívající AJAX se odešle požadavek na webový server na pozadí (asynchronně), zatímco uživatel aplikaci dále používá. Aplikace se tak uživateli jeví mnohem více interaktivní.

V aplikaci využívající AJAX zpracuje kód v JavaScriptu odpověď od serveru a prezentuje ji uživateli. V kombinaci s kaskádovými styly (CSS) a vhodným designem poskytuje taková aplikace skvělou užitnou hodnotu a zachovává si takovou portabilitu, jakou mohou nabídnout pouze webové aplikace.

Základním stavebním kamenem pro vytváření aplikací v AJAXu je objekt XMLHttpRequest. Zatímco mnoho aspektů bylo standardizováno v rámci standardu ECMAScript a konsorciem W3C, tak objekt XMLHttpRequest nikdy standardizován nebyl. Od verze Internet Exploreru 7 je vytváření tohoto objektu ve všech prohlížečích stejné. Pokud ale má správně pracovat i v nižších verzích Internet Exploreru, pak musí být vytvářen jako ActiveX objekt.

2.5 CSS [3]

Původně záměrem jazyka HTML byl popis struktury dokumentu a popis vazeb mezi nimi. Tedy určit v dokumentu, která jeho část je nadpis, která obrázek, číslovaný seznam nebo citace. Ale vlastní vzhled dokumentu měl být určen jinde. Potřeby autorů a zadavatelů se však rychle navyšovaly a výrobci prohlížečů se postarali o vývoj nových prostředků, doplňků a rozšíření jazyka HTML nad rámec standardu.

S přibývajícími nestandardními rozšířeními se z dokumentu začala postupně vytrácet jejich struktura a kód se stále více jen zaměřoval na popis výsledného vzhledu dokumentu. Nakonec se pro pozicování prvků dokumentu začaly používat jako mohutný nástroj tabulky a struktura dokumentů HTML se vytratila úplně.

Před několika málo lety však na síle začala nabírat standardizační skupina W3C (World Wide Web Consortium), která začala zastřešovat projekty, které souvisely s publikováním na webu a začaly řídit nové sjednocující standardy, díky nimž se trend nestrukturovaných HTML dokumentů začíná vytrácet. Vývoj formátu dokumentu HTML byl ukončen a začíná se nahrazovat relativně novým formátem XHTML, ve kterém jsou všechny formátovací značky, a podobná rozšíření, zcela vypuštěny.

Pro formátování dokumentu se tak začala používat technologie jazyka Tabulek kaskádových stylů (CSS). Tento jazyk umožňuje formátovat dokumenty, definuje způsob jejich prezentace na koncových zařízeních, popisuje podobu jednotlivých stránek a styl jednotlivých prvků, a přitom nijak neovlivňuje obsah dokumentů samotných.

Kaskádové styly se ve své podstatě mohou používat v zásadě dvěma způsoby. Buď jako doplněk "starého" formátování pomocí značek HTML, i jako plnohodnotný formátovací nástroj. První přístup je snadnější a dnes také nejvíce rozšířený. Rozšířením stávajících stránek o CSS můžeme odpoutat náš grafický rozlet, avšak všechny uvedené nevýhody se tímto neodstraní. Dokumentu stále chybí struktura, kód je málo přehledný, složitěji se upravuje a udržuje a stále obsahuje přebytek zbytečných značek. Je však vryt do způsobu práce většiny

webových vývojářů a těžko se odstraňuje, protože se web takto dělá mnoho let. Nejprve se vymýšlí, jak stránka bude vypadat, a tomu se následně přizpůsobuje kód, někdy i obsah samotný.

Přístup k tvorbě stránek, tak jak je zamýšlen standardy (X)HTML a CSS je úplně jiný, odlišný v samotném principu. Při návrhu dokumentu se předem vůbec nezvažuje jeho vzhled, ale postup je úplně opačný od postupu prvního přístupu k tvorbě. Vychází se z obsahu, který má dokument sdělit a nejlépe jej strukturovat - určit pořadí částí dokumentu, jejich hierarchii, vzájemné vazby a vhodně označit typizované úseky. Zásadním cílem tedy je, aby dokument začínal těmi nejdůležitějšími informacemi a postupoval dále k informacím méně důležitým, tak aby jeho struktura co nejlépe odpovídala sdělovanému obsahu.

3 Zabezpečení [4] [5]

Zabezpečení webového sídla začíná jeho provozováním na zabezpečeném webovém serveru. Nastavení a udržování jeho zabezpečení vyžaduje důkladnou znalost použitého operačního systému. Před samotným zabezpečením je vhodné se ujistit, že operačním systém je aktuální a byly u něj aplikovány všechny nutné opravy. Proces zabezpečování serveru nazývaný hardening, neboli opevňování, je hlavní věcí při vytváření zabezpečeného serveru. Při tomto procesu by se měly, krom jiného, odstranit všechny služby, které nejsou využívány. Například při vzdálené správě UNIX systému je vhodné službu telnet nahradit zabezpečenou službou SSH nebo OpenSSH. Tato služba také snižuje potřebu použití FTP. U dalších služeb je pak vhodná úvaha o přesunu na jiné počítače.

Vhodné je také sledování souborů protokolů serveru. Při odchylce od normálního vzhledu protokolu je něco špatně. Udržování aktuálnosti operačního systému a všech oprav je další důležitou částí správy zabezpečeného serveru.

3.1 Zabezpečení webového serveru

Zabezpečení webového serveru se skládá z několika kroků. Jelikož je v úvaze, že psaný software bude využívat webového serveru Apache, bude tato kapitola zaměřena na něj. Většina principů je však stejná, více či méně, i pro jiné webové servery.

3.1.1 Práva k adresářům ServerRoot

Pro *ServerRoot*, tedy adresář, ve kterém je Apache nainstalován, by neměl být vlastníkem uživatel, pod kterým je webový server provozován (např. *www* nebo *nobody*). Dále by normální uživatelé i uživatel, pod kterým běží webový server, neměli mít možnost cokoliv měnit v *ServerRoot* nebo v adresářích pod ním. Tímto způsobem tedy lze zajistit, aby bylo dosti obtížné, či téměř nemožné, aby uživatelé měnili nastavení webového serveru. V *DocumentRoot* pak lze nastavit, aby uživatelé mohli měnit soubory, ale nic víc. Většinou se ještě pro uživatele, pod kterým webový server běží, nastavuje právo zápisu do adresáře *log*, pro získávání protokolů.

3.1.2 Znemožnění obcházení nastavení

V Apache se tento problém řeší pomocí direktivy *AllowOverride*. V této direktivě se zadá parametr None pro kompletní znemožnění obcházení nastavení uživateli, nebo lze nastavit, která nastavení překročit mohou.

3.1.3 Ochrana souborů

Webový server by neměl mít přístup k souborům mimo *DocumentRoot*. To lze zabezpečit pomocí *Directory*:

```
#       <Directory />
#               AllowOverride None
#               Options None
#               Order deny, allow
#               Deny from all
#       </Directory>
```

A následně opět pomocí *Directory* povolit přístup k souborům právě v *DocumentRoot*:

```
#       <Directory /usr/local/apache/htdocs>
```

```
#               AllowOverride None
#               Options Indexes FollowSymlinks
#               Order allow, deny
#               Allow from all
#       </Directory>
```

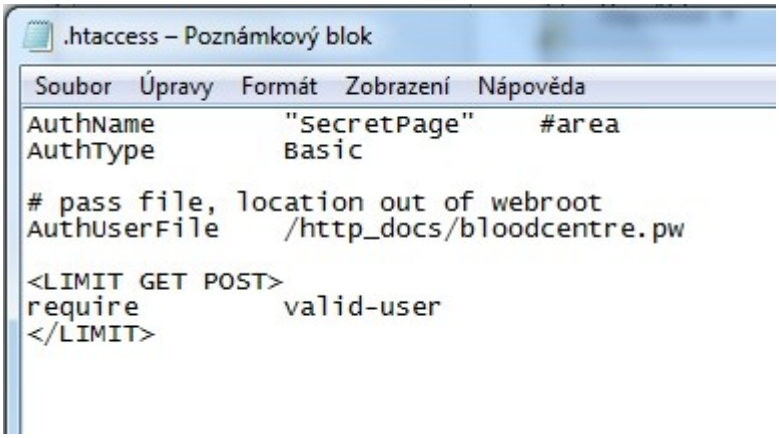
3.2 Identifikace a ověřování uživatelů

Oblast uživatele obsahuje důvěrná data respektive data, ke kterým by měl mít přístup pouze vlastník daného profilu. Proto úvodní strana je zároveň strana autentizační. Ovšem pro zvýšení zabezpečení celého systému byl zvolen způsob registrace uživatele, který je odlišný od většiny, na internetu viditelných, registračních protokolů. Registraci si v tomto případě nebude zakládat uživatel sám, ale založí jej uživatel-administrátor v administrátorském účtu. Částečně se tím předejde bezúčelnému registrování uživatelů. Jelikož je tento software navrhován pro krevní centra, tak celá registrace bude vypadat následně. Při osobní návštěvě pacienta bude pouze sestra, resp. osoba s administrátorským účtem, schopna registrovat nový uživatelský účet. Pacientovi bude přiděleno jeho identifikační číslo a heslo pro vstup do uživatelské části. V ní si již pacient bude schopen heslo sám změnit.

3.2.1 Ověřování webovým serverem

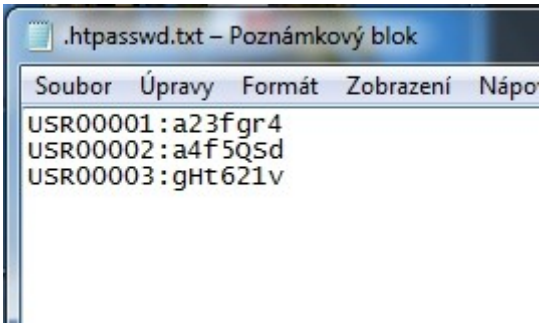
Toto je standardní způsob ověřování uživatelů, o který se stará sám Apache. Níže uvedené direktivy (Obrázek 1) se vkládají do souboru *.htaccess* v chráněném adresáři. Následně programem htpasswd, který je součástí Apache, se vytvoří soubor se jmény a hesly, a umísťuje se mimo strom webu s přístupem pouze pro vlastníka a právem ke čtení pro webový server. Při pokusu prohlížeče o přístup k souborům je vygenerováno přihlašovací okénko. Výhodou také je, že při náhodném získání souboru s hesly cizí osobou, je tento soubor šifrován.

Apache podporuje i ověřování s výtahy hesel, což znamená, že prohlížeč zasílá heslo v zašifrované podobě. I když tato možnost zvyšuje zabezpečení, je ji možno použít pouze s prohlížečem Internet Explorer, což je dnes, při masivním vzrůstu používání i jiných prohlížečů, nevýhoda.



Obrázek 1 Editace souboru .htaccess

Soubor .htaccess je konfiguračním souborem serveru. Využívá se nejčastěji pro vytvoření vlastních chybových stránek či zablokování výpisu složky. Jedinou podmínkou použití .htaccess je jeho podpora ze strany serveru. Vytvořit jej lze v běžném poznámkovém bloku. Jelikož se jedná o skrytý soubor, je nutno mít zapnuto zobrazení skrytých souborů a složek v průzkumníku.



Obrázek 2 Možná podoba souboru .htpasswd

3.2.2 Identifikace a ověřování uživatele v PHP

Ověřování uživatele v PHP je o něco složitější než pouhé zabezpečení přístupu na straně serveru, avšak má pár výhod, které v Apache možné není. Může být zrušeno, to znamená, že uživatel se může ze stránek odhlásit. Toto v Apache zabezpečit nelze, stejně tak jako časové odhlášení, které může zabezpečit odhlášení uživatele např. při jeho neaktivitě na stránkách. Tento způsob by mohl být přizpůsoben například pro menší Java applet, který by odeslal zakódované heslo přes internet a pomocí mcrypt knihovny by mohlo být zpětně dekódováno. Další výhodou je možnost založení na databázi. Tedy používání menších kousků dat z databáze pro ověřování. Přístup se také může podle stránky měnit, tedy že k některým stránkám bude uživatel ověřován a k jiným zase ne. Apache v tomto není tak flexibilní jako PHP. Neposlední výhodou je také práce s PHP ve verzi CGI. To je velké plus.

3.2.3 Kontrola IP adres

V docela častém povědomí lidí je, že adresa IP identifikuje návštěvníka jednoznačně. Proxy servery však mohou způsobit, že požadavky více uživatelů přijdou ze stejné IP adresy, tedy by se v podstatě kontrolovala pouze adresa proxy serveru, což je nechtěné. Dalším problémem tohoto způsobu zabezpečení je i to, že uživatel může přistupovat na stránky z více IP adres. V nástinu na téma této práce, že dárce by se mohl přihlašovat na odběr pouze z domu a z práce nebo školy již ne. To by bylo dost omezující a pro uživatele nevhodné. Díky tomu je identifikace uživatelů podle IP adres dosti obtížná a pro tuto práci nevhodná.

3.3 Kryptografie [6]

Kryptografie se v PHP nejčastěji používá pro kódování informací a generování kontrolních součtů. Její používání, při správném použití, umožňuje zvýšit úroveň zabezpečení. V PHP je většina kryptografických funkcích dostupná v knihovnách mcrypt a mhash. Tyto knihovny se pak musí nainstalovat a nastavit v PHP, aby je používalo, pomocí *-with-mcrypt*

a *-with-mhash*. PHP verze 3.0.12 a starší může při kompilaci s podporou mcrypt dělat problémy, jelikož soubor knihovny mcrypt není kompletně synchronizován s aktuální verzí knihovny mcrypt. To v dnešní době však může být ojediněle, protože aktuální verze PHP je 5.3.9.

3.3.1 Používání kódování

Jsou v podstatě dvě možnosti kryptografie. S tajným nebo veřejným klíčem. Někdy používané označení symetrický a asymetrický. Kryptografie s klíčem veřejným je označována za asymetrickou z důvodu, že klíče pro kódování a dekódování zprávy nejsou stejné. Tento způsob se například může použít v situaci, kdy ostatní uživatelé mohou posílat zprávy zakódované veřejným klíčem, ale k odkódování a přečtení zprávy je zapotřebí privátní klíč, který může vlastnit jen jedna osoba. Z trochu opačné strany, tento typ kódování může být použit i pro ověření, že zpráva byla kódována privátním klíčem a lidé s veřejným klíčem ji mohou přečíst. Problémy kryptografie s veřejným klíčem jsou velké klíče (512 - 2048 b) a nízká rychlost.

Kryptografie s klíčem tajným je označována za symetrickou z důvodu, že klíče pro kódování a dekódování jsou stejné. Z tohoto důvodu by klíč měl být udržován v absolutní tajnosti. Tento typ kryptografie je rychlejší a používá menší klíče (40 - 128 b). Obvykle se v programech používají veřejné klíče pro podepisování zpráv a odesílání tajných klíčů.

Pro potřeby kódování aplikací se většinou uchyluje k použití kryptografie s tajným klíčem. K dispozici je celá řada algoritmů, z nichž některé jsou považovány za nedostatečně chránící (RC2, DES) a některé zase za velmi bezpečné (Blowfish, RC5, IDEA). Zabezpečení poskytnuté hostitelským algoritmem závisí na velikosti klíče. Rozluštění zprávy kódované 128 bitovým klíčem je prakticky nemožné.

Kódovací algoritmy lze použít ve 4 režimech:

Electronic Code Book (ECB) - je nejlépe použitelný nástroj pro kódování dat malých velikostí, která obsahují víceméně náhodný obsah. Například jiné kódovací klíče. V režimu ECB je každý blok dat kódován zvlášť. Jakým způsobem tento režim pracuje, je lepší se mu vyhnout a zvolit režim CBC, který je podstatně bezpečnější.

Cipher Block Chaining (CBC) - je režimem, který je používán nejčastěji. Je většinou nejčastější volbou v situaci, kdy programátor neví, který režim přesně použít.

Cipher Feedback (CFB) - Je určen pro kódování bajtových datových proudů.

Output Feedback (OFB) - Jeho použití je obdobné použití režimu CFB.

3.3.2 Kódovací funkce Hash

Velkou většinu kódovacích algoritmů lze použít pro generování zakódované podoby nebo výtahu zprávy. Jako příklad tohoto kódování je hodnota CRC, kterou používají komprimační algoritmy pro kontrolu případného poškození archivu. Kódování se často používá pro ukládání hesel. Při uložení hesla v zakódované podobě, se heslo nemusí ověřovat, když se někdo přihlašuje bez uloženého aktuálního hesla na serveru. Ve skriptech PHP je pro vygenerování zakódované podoby hesla možné použití funkce *crypt*. Pro potřebu kódování většího množství dat se pak používá modul *mhash*. Při použití této funkce je pak potřeba PHP zkompilovat s podporou *mhash*. Výstupem funkce *mhash*, stejně jako *mcrypt*, jsou binární data a pro jejich další použití, např. jako cookies, je nutné je převést do hexadecimálního tvaru funkcí *bin2hex*.

3.4 Zabezpečený přenos pomocí SSL

Použití webového serveru s podporou SSL (Secure Socket Layer) je skvělý způsob rozšíření a zlepšení zabezpečení webového sídla bez změny řádku kódu. SSL používá kryptografii pro ochranu toku informací mezi webovým serverem a prohlížečem. SSL všechna data procházející internetem kóduje a zároveň ověřuje obě strany. Vlastnosti SSL jej dělají velmi vhodným pro použití v aplikacích, kde dochází k přenosu citlivých dat.

SSL používá kryptografii s veřejným klíčem (viz. kapitola Používání kódování), kde server při spojení odešle klientovi veřejný klíč pro zakódování informací, které může dekódovat pouze server svým soukromým klíčem. Zabezpečené HTTP se pak obvykle liší číslem portu od HTTP nezabezpečeného (443 místo 80), což umožňuje serveru sdělit rozdíl a příslušně odpovědět.

3.5 Bezpečné PHP skripty

Existují postupy programování, které zvyšují bezpečnost PHP skriptů. Spouštění skriptů PHP je mnohem bezpečnější než spouštění skriptů CGI, ale přesto se zde může vyskytovat dost věcí, které mohou fungovat špatně. Aktivovaný zabezpečený režim může omezit důsledek toho, že něco nefunguje tak, jak má. Aplikace založené na webu často běží delší dobu bez důkladnějšího dozoru. O problému se ve většině případech dozví jako první uživatelé aplikace. Proto by měl být zřízen způsob, aby uživatelé mohli vlastníka (vývojáře) o problému informovat. Vyvarování by se také mělo týkat odesílání citlivých informací skrze internet takovým způsobem, kdy by je mohl získat někdo jiný, např. pomocí GET, POST, cookies nebo zakódováním informací do URL. Použití SSL serveru je nejspíše nejjednodušším a nejlepším způsobem, protože šifruje celou komunikaci mezi prohlížečem uživatele a serverem.

4 MySQL Databáze

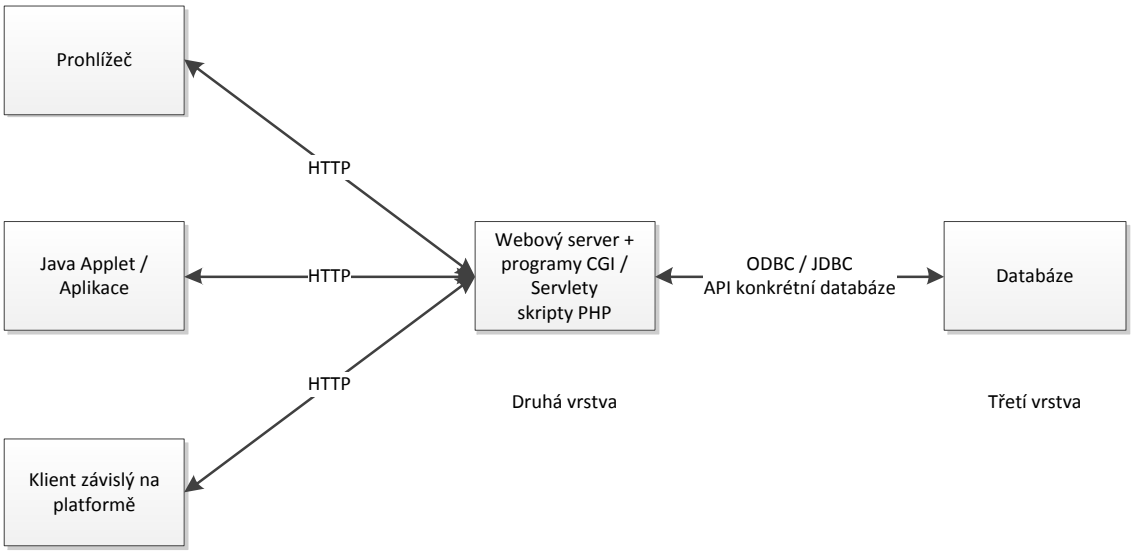
4.1 Architektura webové databázové aplikace [4]

Základními částmi nebo vrstvami databázové webové aplikace jsou:

- klient: uživatelův webový prohlížeč, java applet, aplikace java nebo na platformě závislý klientský program
- logika aplikace: zakódovaná do algoritmů používaných ve skriptech CGI, speciálních modulech webového serveru nebo v serveru závislém na aplikaci.
- databázová konektivita: API databáze nebo obecné propojovací protokoly (ODBC, JDBC).
- databázový server: RDBMS, ODBMS

Implementace takových aplikací může být provedena pomocí vícevrstvého modelu z důvodu jedné nebo více vrstev spojených dohromady. Obvyklou implementací je třívrstvý systém:

1. vrstva - webový klient
2. vrstva - webový server, CGI skripty a API pro připojení k databázím
3. vrstva - databázový server



Obrázek 3 Třívrstvý systém

4.2 Podpora API MySQL v PHP

Práce s MySQL se zahajuje připojením k serveru pomocí MySQL_Connect(). Parametrem funkce je jméno počítače, na kterém MySQL běží. Pro databázi chráněnou jménem a heslem jsou pak parametry včetně jména počítače i jméno a heslo uživatele. Výsledkem této funkce pak je číslo, které identifikuje připojení k serveru a je dál používáno v dalších funkcích. [4]

Následně je zapotřebí výběr používané databáze. K tomuto slouží funkce MySQL_Select_DB. Parametry této funkce jsou jméno databáze a číslo spojení. Číslo spojení

se využívá v situacích, kdy skript pracuje s více spojeními s více databázemi. Pokud však pracuje pouze s jednou, lze číslo spojení v parametrech vynechat. [4]

Po připojení k MySQL serveru a zvolené databázi, pak lze používat funkci MySQL_Query(), pomocí které se kladou dotazy, v jazyce SQL, na databázi. Prvním parametrem funkce je SQL dotaz, druhým pak číslo spojení. Výsledkem této funkce je číslo identifikující výsledek. Pomocí tohoto čísla se pak na výsledek odvolává v dalších funkcích. [4]

Po získání výsledku dotazu se následně pro čtení jednotlivých výsledků používají dvě časté funkce. Jednou z nich je MySQL_Fetch_Array(), která přečte jeden záznam a uloží jej do asociativního pole. [4] Záznam je asociován s názvem sloupce v tabulce databáze. Asociativní pole je proměnná, ve které se nepoužívají celočíselné indexy pro určení pozice prvku v poli, ale indexy jsou řetězce.

4.2.1 Vkládání dat do InnoDB pomocí PHP [12]

Rozdíly rychlosti vkládání do InnoDB vzhledem k různě zapsaným možnostem. Vykonání níže uvedeného kódu trvalo 26,448 s.

```
<?php
for ($i=0; $i < 1000; $i++) {
          mysql_query("INSERT INTO tab VALUES ($i)");
}
?>
```

Oproti tomu vykonání kódu

```
<?php
$values = array();
for ($i=0; $i < 1000; $i++) {
            $values[] = "($i)";
}
mysql_query("INSERT INTO tab VALUES " . implode(", ", $values));
?>
```

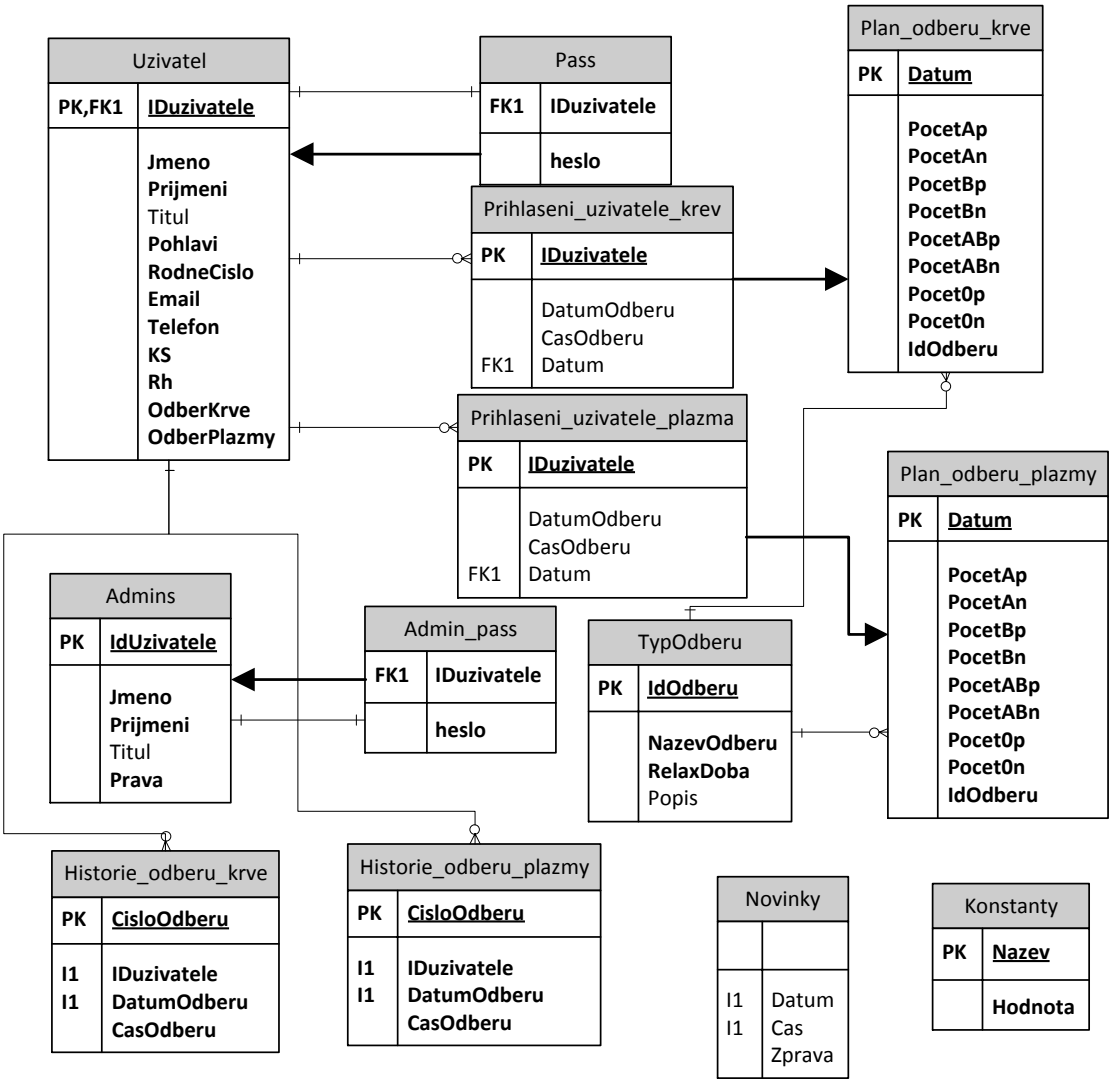
trvalo 0,035 s.

Bez transakcí jsou InnoDB při vkládání záznamů o dva řády pomalejší než MyISAM. Příčinou je nastavení direktivy "innodb_flush_log_at_trx_commit" na výchozí hodnotu 1, které způsobí synchronizaci dat na disku po každé transakci, což je v autocommit režimu po každém příkazu. Při uzavření všech vložení do jedné transakce se výkon mnohonásobně zlepší, stále ale je pomalejší než u tabulek MyISAM.

4.3 Stavba databáze

Zjednodušený datový model je zobrazen na Obrázek 4.

Při návrhu byla zvolena relační databáze obsahující šest tabulek. V tabulce *uzivatel* je primárním klíčem identifikační číslo uživatele - "IDuzivatele". Bylo by možné jako unikátní hodnotu používat i rodné číslo, z důvodu bezpečnosti je každému uživateli vygenerováno náhodné identifikační číslo. Dále jsou v tabulce *uzivatel* umístěny indexované hodnoty krevní skupiny a Rh faktoru. Bylo by možné při nedostatku určité krevní skupiny e-mailem zvát dárce. Důležitou položkou v tabulce je položka "MozneOdbery", která obsahuje identifikační čísla odběrů z tabulky *TypOdberu* a podle nichž se dynamicky generují možnosti přihlášení pro dárce. Podřazenou tabulkou pro tabulku *Uzivatel* je tabulka *Pass*, ve které je každému uživatelskému jménu přiřazeno vstupní heslo.



Obrázek 4 Databázová struktura

4.3.1 Entity a vztahy mezi nimi

Uživatel - tato entita obsahuje identifikační číslo uživatele a krom dalších je zde také zahrnuta krevní skupina, která se na začátku programu uloží do relace, a podle ní jsou dynamicky vytvářeny stránky v uživatelském prostředí.

Pass - je entita obsahující soubor hesel každého uživatele. Kardinalita mezi entitou Uživatel a entitou Pass je 1:1 a každý uživatel má přiřazeno právě jedno heslo. Silnější šipka ukazuje na nadřazenou tabulku. Relace mezi nimi jsou: při aktualizaci nebo odstranění nadřazené položky se kaskádovitě odstraní položky podřazené.

Admins - obsahuje základní údaje a Id uživatelů s administrátorskými a organizačními právy.

Admin_pass - obsahuje soubor hesel pro uživatele s administrátorskými a organizačními právy.

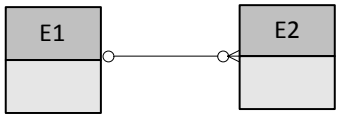
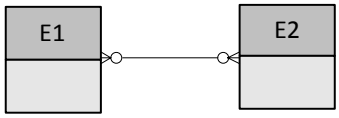
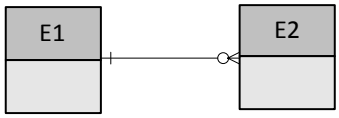
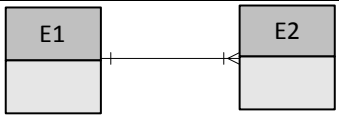
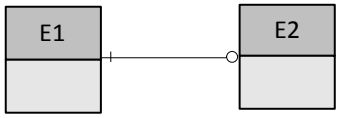
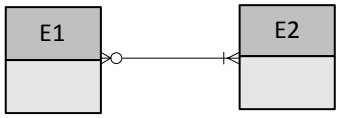
Plan_odberu_krve a **Plan_odberu_plazmy -** jsou entity obsahující počty dárců jednotlivých krevních skupin na jedno určité datum. Jsou to nadřazené entity pro entity Prihlaseni_uzivatele_krev a Prihlaseni_uzivatele_plazma s relacemi při aktualizaci nebo odstranění položky nadřazené se kaskádovitě odstraní položky podřazené.

Prihlaseni_uzivatele_krev a **Prihlaseni_uzivatele_plazma -** entity obsahující data a hodiny odběrů, na které jsou dárci již přihlášení. Aby nedošlo k chybě, kdy při zrušení odběrového dne na něj zůstali uživatelé přihlášení, je zajištěno relací s nadřazenými tabulkami Plan_odberu_krve a Plan_odberu_plazmy, kaskádovitým odstraněním.
Po uplynutí odběrového dne je v Linuxovém správci úloh Cron naplánována úloha, MySQL příkaz, kdy pokud jsou řádky daného data s příznakem Dostavil - 'A', přesunuty do tabulky Historie_odberu_krve nebo Historie_odberu_plazmy. Pokud mají příznak Dostavil - 'N', pak jsou tyto řádky z tabulky odstraněny.

Historie_odberu_plazmy a **Historie_odberu_krve -** entity obsahující všechny proběhlé odběry všech dárců.

Konstanty - Entita obsahující všechny potřebné konstantní hodnoty, které lze v aplikaci aktualizovat.

Novinky - Entita pro ukládání novinek zobrazujících se na úvodní straně aplikace.

Zobrazení	Kardinalita	Parcialita
	1:N	Záznam z E1 se může vázat k více záznamům z E2. Záznam z E2 se může vázat maximálně k jednomu záznamu z E1.
	N:M	Záznam z E1 se může vázat k více záznamům z E2. Záznam z E2 se může vázat k více záznamům z E1.
	1:N	Záznam z E1 se může vázat k více záznamům z E2. Záznam z E2 se musí vázat právě k jednomu záznamu z E1.
	1:N	Záznam z E1 se může vázat k více záznamům z E2, minimálně však k jednomu. Záznam z E2 se musí vázat právě k jednomu záznamu z E1.
	1:1	Záznam z E1 se může vázat maximálně k jednomu záznamu z E2. Záznam z E2 se musí vázat právě k jednomu záznamu z E1.
	N:M	Záznam z E1 se může vázat k více záznamům z E2, minimálně však k jednomu. Záznam z E2 se může vázat k více záznamům z E1.

Tabulka 1 Použité značení v databázovém modelu

4.4 Bezpečnost komunikace s databází [4] [5] [7]

4.4.1 SQL Injection

SQL Injection útok (SQL Insertion útok), je podskupinou útoků "code insertion". Jedná se o techniku napadení databázové vrstvy programu vsunutím (odtud "injection") kódu přes neošetřený vstup a vykonání vlastního, většinou pozměněného, SQL dotazu. Toto nechtěné chování vzniká při propojení aplikační vrstvy s databázovou vrstvou, protože se téměř vždy jedná o dva různé programy, a zabraňuje se mu pomocí jednoduchého opatření speciálních znaků zpětnými lomítky.

4.4.1.1 **Obrana na straně aplikace** [7]

V PHP existuje několik způsobů, jak se bránit SQL Injection. Některé databázové vrstvy umožňují oddělení SQL dotazů od uživatelských dat, které jsou předávány jako parametry. Pokud server takovou vrstvou nedisponuje nebo se bez ní musí programátor obejít, je nutné parametry SQL dotazů předávat přímo jako jejich součást. A na tomto místě nastupuje obrana proti SQL Injection. Jeden ze způsobů je ošetřit veškerá vstupní data vhodným způsobem, druhý je použít direktivu **magic_quotes_gpc** a všechny hodnoty uzavírat do apostrofů.

```
<?php
// ošetření vstupních hodnot
MySQL_Query(SELECT * FROM tabulka WHERE nazev = '" .
MySQL_real_escape_string($_GET["nazev"]) . "' OR id = " .
intval($_GET["id"]));
// použití magic quotes gpc
MySQL_Query(SELECT * FROM tabulka WHERE nazev = '$_GET[nazev]' OR id =
'$_GET[id]');
?>
```

Při použití magic_quotes_gpc je do apostrofů uzavřená i číselná hodnota. MySQL to umožňuje a pokud předaný řetězec nebude na číslo převoditelný, převede se na nulu. Druhý způsob má dost programátorů v oblibě, protože umožňuje vytvářet jednoduchý a přímočarý kód. Pokud se ale na nastavení této direktivy nelze spolehnout, pak nelze použít ani jeden ze způsobů. Druhý způsob nelze použít kvůli riziku SQL Injection a první z důvodu, že kdyby byla direktiva zapnutá, tak se speciální znaky v předaných datech budou zpětnými lomítky opatřovat dvakrát - jednou kvůli direktivě magic_quotes_gpc jednou funkcí addslashes. Tento případ se pak řeší pouze vytvořením vlastní funkce a jejím použitím místo funkce addslashes.

V některých případech je nutné předaná data ošetřit i v případě zapnuté direktivy magic_quotes_gpc.

```
<?php
// nefunguje
MySQL_Query("SELECT * FROM tabulka LIMIT 10 OFFSET '$_GET[offset]'");
// funguje
MySQL_Query("SELECT * FROM tabulka LIMIT 10
OFFSET".intval($_GET["offset"]));
?>
```

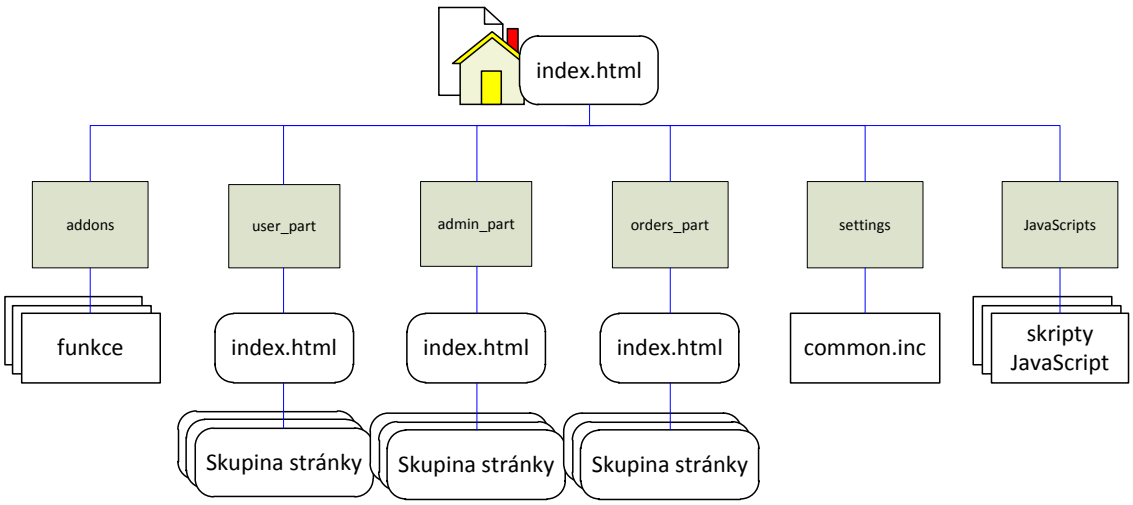
4.4.1.2 Obrana na straně databáze

V databázi lze útokům zabránit, nebo jej alespoň útočníkovi ztížit, vhodným nastavením práv uživatele, se kterými bude program přistupovat. Není třeba přímo z aplikační vrstvy mazat tabulky či celé databáze, proto stačí povolit pouze základní SQL příkazy.

5 Stavba informačního systému

5.1 Adresářová struktura

Úvodní stránka index.html je tvořena přihlašovacím formulářem. Podle přihlašovacích údajů je pak uživatel přesměrován do jednoho z adresářů (user_part, admin_part, orders_part) podle toho, jaká přístupová práva má. V těchto adresářích se nachází pouze základní koncept aplikace. Základní parametry nastavení jsou uvedeny v adresáři settings a parciální funkce tvořící jádro aplikace jsou umístěny v adresáři addons. JavaScripty a CSS mají své umístění v adresáři JavScripts a Style.



Obrázek 5 Adresářová struktura aplikace

5.2 Autentizace a autorizace

Autentizace je proces ověření proklamované identity subjektu. Po dokončení procesu autentizace pak následuje proces autorizace, což je souhlas, schválení a umožnění přístupu či provedení konkrétní operace daným subjektem. Patří k bezpečnostním opatřením a zajišťuje ochranu před falšováním identity. [8] Toto prověření se provádí na přihlašovací stránce a je vytvořeno na základě uživatelského identifikačního čísla, které je každému uživateli vygenerováno při jeho vložení do databáze dárců a hesla, které je zprvu taktéž náhodně vygenerováno, avšak po přihlášení je dárci umožněna jeho změna. Přihlašovací stránka (Obrázek 6) je tak jediná dostupná pro nepřihlášené uživatele.

Obrázek 6 Výřez přihlašovací obrazovky

Po procesu autentizace následuje proces autorizace, při kterém jsou uživatelům přidělována práva. Tento proces probíhá ihned po přihlášení a při každém novém načtení stránky v programu. V případě, že by se uživatel po přihlášení pokusil pozměnit url a dostat se do části aplikace, ke které nemá práva, pak je automaticky přesměrován zpět na přihlašovací stránku.

Na Obrázek 7 je uveden postup autentizace a autorizace.

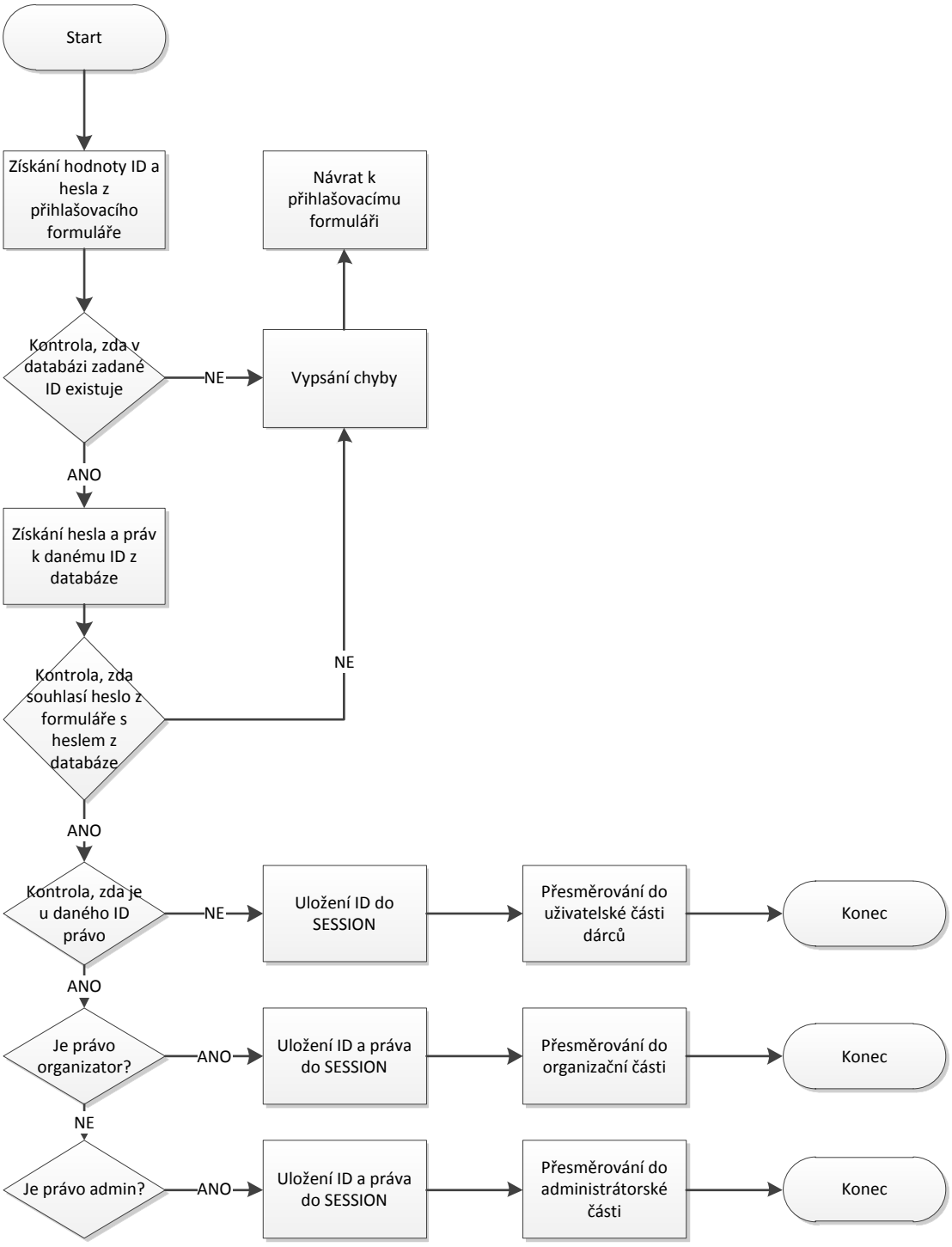
5.2.1 Používání relací (SESSION)

U složitějších webových aplikací je potřeba si uchovávat některé informace mezi načteními jednotlivých stránek. Pro toto byly navrženy cookies, které jsou samy o sobě snadno zranitelné. Pokud by byla nastavena cookie identifikující přihlášeného uživatele, nic by útočníkovi nebránilo v tom, aby si hodnotu této cookie měnil podle potřeby. Cookie navíc mají velikostní limit (20 cookies na doménu - tzn. 4 KB na cookie). [9]

PHP má pro ukládání takových informací mechanismus session proměnných, který cookies obvykle také používá, ale pouze pro uchování neuhodnutelného Session ID. Je zde také ovšem riziko, a to, pokud by server s více uživateli byl špatně zabezpečený, mohou ostatní uživatelé Session ID i s daty. Se správou session je spojeno několik různých útoků. Pokud někdo získá cizí Session ID a použije jej pro sebe, pak se jedná o Session Hijacking. Na serveru se tomuto dá zabránit jeho zabezpečením. Na cestě nebo straně klienta už to je horší. Pro ochranu cesty je proto zvolena nejlepší možnost zabezpečení - používání protokolu HTTPS místo HTTP. U klienta jsou možnosti velmi omezené. [9]

V aplikaci pro krevní centrum je nejdříve založena relace pomocí příkazu *session_start()*. Ihned poté je volána funkce *session_regenerate_id()*, která způsobí změnu Session ID.

Tato funkce slouží jako obrana před nežádoucím útokem typu Session Fixation. Jedná se o útok, kdy útočník nastaví nějakou hodnotu Session ID a jakmile se uživatel přihlásí, tak tuto Session ID použije pro sebe. Volání této funkce se většinou provádí před prováděním citlivých operací jako je právě přihlašování.



Obrázek 7 Proces autentizace a autorizace

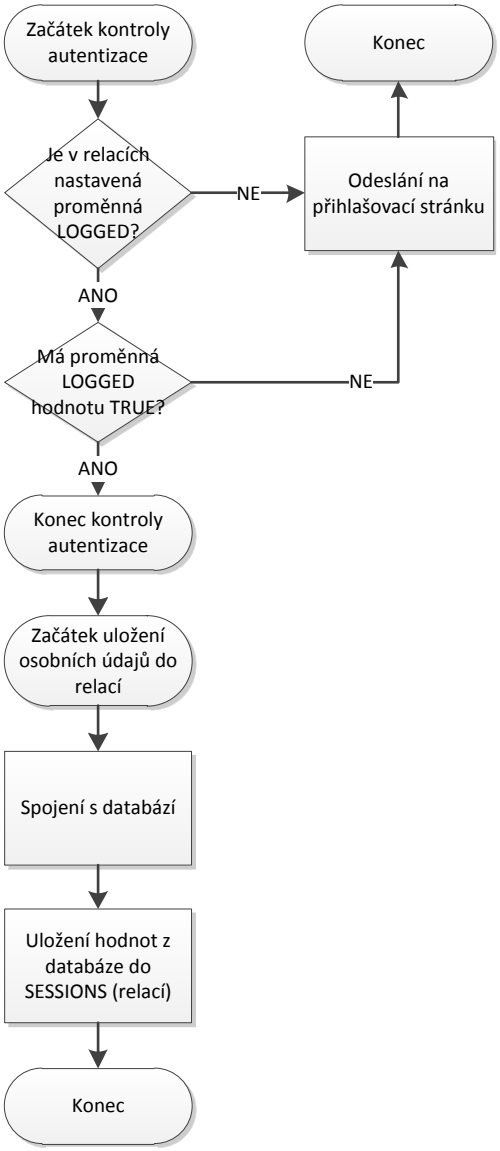
6 Část přístupná pro dárce

V této části má uživatel několik možností:

1. Přihlášení / odhlášení odběrů
2. Zobrazení osobních údajů
3. Editace kontaktních údajů (tel. čísla, emailové adresy)
4. Změna hesla pro vstup do aplikace

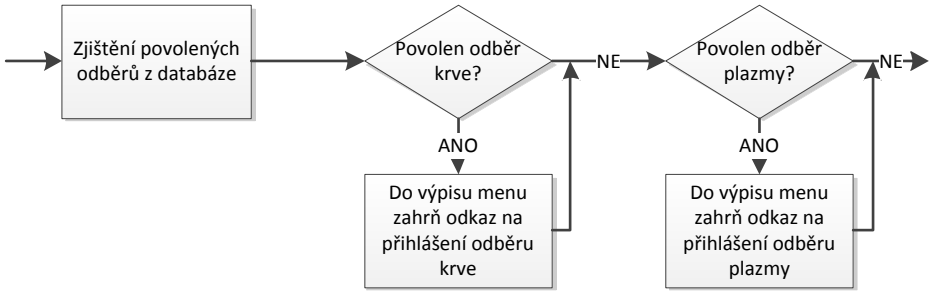
Po autentizaci, autorizaci a následném přesměrování do adresáře user_part je načtena úvodní strana index.html. V ní jako první proběhne kontrola autentizace. Pokud uživatel neprošel autentizačním a autorizačním procesem, je odkázán zpět na přihlašovací formulář (viz. Obrázek 8).

Další PHP skript, který ihned po jeho přihlášení probíhá je uložení jeho základních informací do relací. Toto je zajištěno z důvodu rychlosti aplikace. Aby bylo zajištěno, že uživatel se nebude moct přihlásit na odběr, který má z jakéhokoliv důvodu nedostupný, je potřeba jej neustále kontrolovat. Každé přihlášení a vyhledávání v databázi je ovšem zbytečné a proto jsou zásadní informace staženy najednou při přihlášení a uloženy do relací (Obrázek 8).

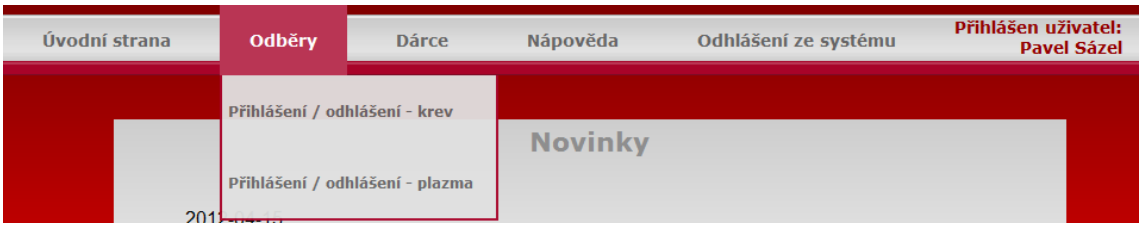


Obrázek 8 Procesy při vstupu aplikace

Vytvoření menu v aplikaci zajišťuje PHP skript, který menu uživateli dynamicky přizpůsobuje. V případě, kdy by uživatel neměl povolen jakýkoliv typ odběru, tak mu jej menu ani nenabídne.



Obrázek 9 Tvorba menu podle parametrů pacienta



Obrázek 10 Menu - povolen odběr krve i plazmy

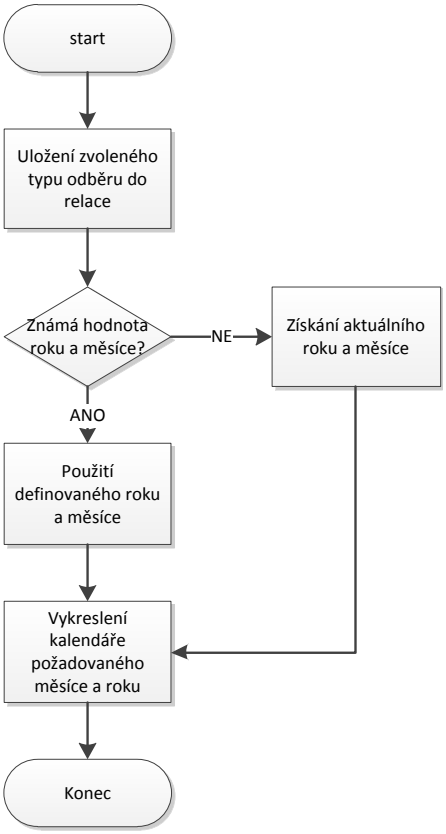
6.1 Úvodní strana

Úvodní stranu tvoří již zmíněné menu a tabulka novinek. Tato tabulka je uživatelem - dárcem, needitovatelná, tedy jen ke čtení, a slouží pro hromadné sdělení podstatných informací ze strany krevního centra k uživatelům. Původně byly všechny novinky umístěny v textovém souboru a na stránky byl vypisován jen obsah tohoto souboru. Bylo tak zamýšleno z důvodu urychlení aplikace. Následně však byl tento soubor nahrazen databázovou tabulkou z toho důvodu, že soubor lze, chtěně či nechtěně, snadno smazat. Časový rozdíl ve vypsání byl více než zanedbatelný. Pro přehlednost je ještě každá zpráva barevně odlišena.

6.2 Přihlášení k odběru / Odhlášení z odběru

Pro usnadnění orientace je v menu přihlášení / odhlášení odběru rozděleno zvlášť pro plazmu a zvlášť pro krev. Obojí je však zajišťováno jedním PHP skriptem. V odkazu na tento skript je odeslán i parametr, který odběr si uživatel zvolil. Z důvodu zvýšení bezpečnosti je hodnota tohoto parametru kódována. Poslední kontrolou v tomto skriptu je kontrola hodnot v databázi, zda uživatel má přihlášení ke zvolenému typu odběru povoleno. Pokud tímto testem neprojde, je vrácen zpět na úvodní stranu. Toto zabezpečení by mělo být dostatečné, jelikož sessions jsou sice uloženy na straně serveru, ne jako cookies v prohlížeči uživatele, ale ověření hodnot přímo v databázi je bezpečnější.

Funkce pro vykreslení kalendáře vyžaduje dva parametry - rok a měsíc. Celý skript na začátku zkontroluje, zda se v adresním řádku nenachází informace o hodnotách pro vykreslovaný rok a měsíc. Pokud tak není, tak vykreslí kalendář pro aktuální rok a měsíc. Při prvním spuštění skriptu se v adresním řádku tyto hodnoty nevyskytují. Je ale potřeba se pohybovat i mimo aktuální měsíc nebo rok. Toto je zajištěno odkazy, které odkazují na totožnou stránku, avšak mají v sobě uloženy hodnoty proměnných pro vykreslovaný měsíc i rok.

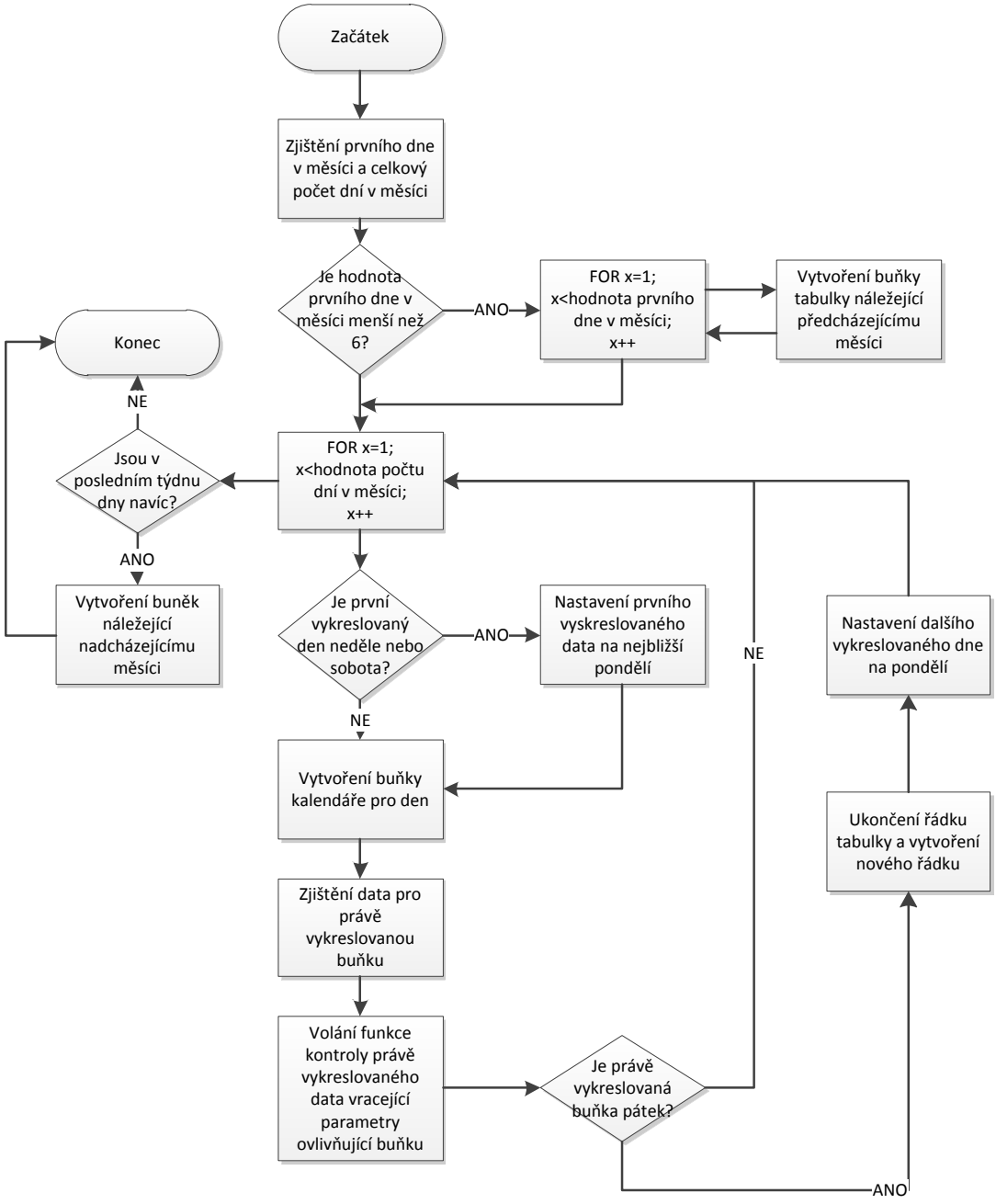


6.2.1 Funkce pro vykreslení kalendáře

Celý kalendář je tvořen generováním textového výstupu HTML kódu. PHP funkce Date(), dokáže vrátit, krom jiných hodnot, také číselnou hodnotu dne v týdnu, kde 0 je rovna neděli a 6 je rovna sobotě, a také číselnou hodnotu počtu dnů v daném měsíci. Funkce pro vykreslení kalendáře si nejdříve zjistí, jakou číselnou hodnotu má první den ve vykreslovaném měsíci a také celkový počet dnů v tomto měsíci. Od těchto dvou hodnot je poté pomocí cyklů FOR vytvořen celý kalendář.

Vykreslována je tabulka o pěti sloupcích, tedy pro každý pracovní den jeden sloupec, a počet řádků je ovlivňován počtem týdnů, které daný měsíc zabírá.

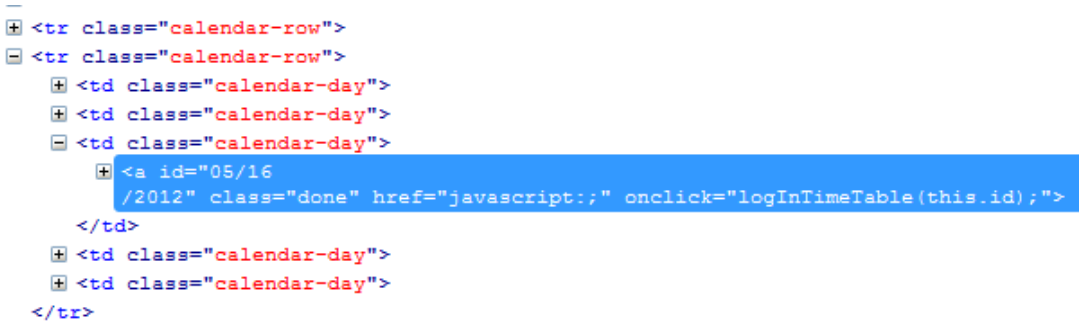
Postup vytvoření kalendáře je znázorněn na Obrázek 11.



Obrázek 11 Vývojový diagram kalendáře

6.2.2 Funkce kontroly generovaného data pro zapsání uživatele k odběru

Při vykreslování buněk tabulky kalendáře pro jednotlivé dny, je do nich vkládán HTML tag odkazu s různými parametry, které jsou probrány dále v textu. Tyto parametry pak ovlivňují další možnosti pro vykreslované datum.



Obrázek 12 Ukázka vygenerovaného HTML kódu a odkazového tagu s parametry

Nejdříve se ověří, zda vykreslované datum předchází datu aktuálního dne. Pokud ano, pak je funkcí kontroly generovaného data vrácen řetězec, který volá funkci javascriptu upozorňující dárce, že právě zvolil datum předcházející aktuálnímu dni a má zvolit datum, které aktuálnímu dni nadchází.

Následně jsou z databáze zjištěna data posledních odběrů krve a plazmy, které předchází aktuálnímu dni. Z nich je vybrán ten odběr, který proběhl jako poslední. Dále jsou vyhledány v databázi data odběrů, která má daný uživatel již naplánována. Všechny tyto data jsou uložena do pole. Souběžně s každým uložením data ať již vykonaného odběru, tak odběru naplánovaného, je zjištěn typ odběru a k tomu z databáze zjištěn počet dní, po které dárce krev darovat nemůže. Tento počet dní je přičten k datu vykonaného nebo naplánovaného odběru a zjištěné datum je uloženo do jiného pole, avšak při zachování stejného indexu, jaký má právě zpracovávané datum odběru.

Pro zajištění toho, aby se dárce nepřihlásil na odběr, jehož následující dny, po které by nemohl darovat, se kryly s již naplánovaným odběrem, je od těchto naplánovaných dat odečten počet dní, po které dárce nebude moct darovat pro odběr, který si zvolil, že by chtěl absolvovat. S hodnotami těchto polí je porovnáváno každé vykreslované datum, a pokud se nachází v některém z rozmezí, a pokud ano, pak je funkcí kontroly generovaného data vrácen řetězec, který při kliknutí na dané datum volá funkci JavaScriptu upozorňující dárce, že právě zvolil datum, na které se přihlásit nemůže, jelikož je již přihlášen na určitý odběr.

Pro lepší pochopení je uveden příklad, k němuž souvisí Tabulka 2.

Současné datum		10.3.2012	
Index	0	1	2
pole Datum odběru	18.3.2012	20.4.2012	23.5.2012
pole Datum povolující odběry	17.4.2012	20.5.2012	22.6.2012
pole Datum zakazující odběry před daným odběrem	3.3.2012	5.4.2012	8.5.2012

Tabulka 2 Hodnoty k vysvětlujícímu příkladu

V databázi bylo zjištěno, že dárce má naplánován odběr na datum 20.4.2012. Jedná se o odběr krve a počet dní, po níž nemůže znova krev darovat je roven třiceti (tato hodnota je

smyšlená pouze pro názornou ukázku). K datu 20.4.2012 je tedy přičteno třicet dní - tedy datum 20.5.2012. Jelikož se dárce nachází v oblasti programu pro přihlášení odběru plazmy, tak počet dní, po které, od tohoto odběru, dárce nemůže darovat je patnáct dní (tato hodnota je smyšlená pouze pro názornou ukázku). Proto patnáct dní před každým již zapsaným odběrem se dárce nemůže přihlásit. Pro datum 20.4.2012 to tedy znamená datum 5.4.2012.

Tyto tři hodnoty tvoří interval, ve kterém se dárce nemůže přihlásit. Těchto intervalů je tolik, kolik odběrů má dárce již zapsaných.

Pokud se právě generované datum kalendáře rovná datu, ve které je uživatel přihlášen k odběru pak je funkcí kontroly generovaného data vrácen řetězec, který při kliknutí na dané datum pomocí technologie AJAX provede další PHP skript pro odstranění uživatele z plánovaných odběrů.

Po této kontrole následuje další kontrola, a to, zda na zrovna vytvářenou buňku tabulky, tedy určité datum, je v databázi zapsán odběr vhodný právě pro přihlášeného uživatele - tedy aby byl vypsán odběr pro krevní skupinu a Rh faktor stejný, jako má právě přihlášený uživatel.

Pokud je odběr vypsán, pak probíhá ještě kontrola, zda již daný termín není plně obsazen. Pokud není, pak je funkcí kontroly generovaného data vrácen řetězec, který při kliknutí na dané datum pomocí technologie AJAX provede další PHP skript pro vykreslení časové tabulky (Obrázek 14).

Obrázek 13 Kalendář s vyznačeným přihlášením k odběru krve a dat, na která se nelze přihlásit

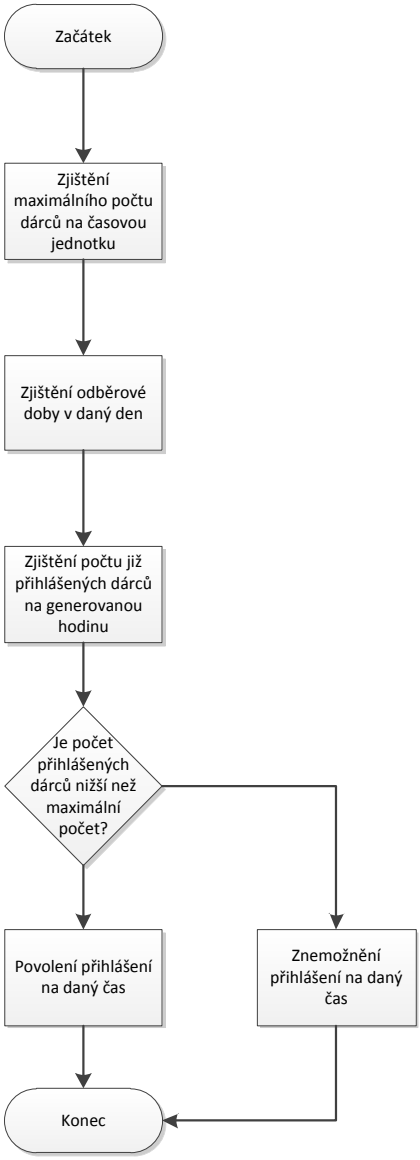
6.2.2.1 Vypsání časů odběrů pro dané datum

Aplikace je navržena tak, že při zapisování termínu odběrů pracovnicí krevního centra, je vyžadován počet odběrů každé krevní skupiny, který by měl být naplněn za jeden celý odběrový den. To, zda je tento termín již obsazen přihlášenými dárci či ne, je kontrolováno již při vykreslování kalendáře odběrů.

Odběr trvá přibližně stejnou dobu a na krevním centru také dokáží odhadnout, kolik dárců se může přihlásit na jednu časovou jednotku. Časová jednotka je v tomto případě přibližná délka odběru. Tuto délku mění uživatel s právem "Admin".

Po zvolení data odběru se tedy objeví časový rozrvh odběrů (Obrázek 14). Každý odběrový den je navíc různě dlouhý. Skript pracuje tak, že si v databázi najde počáteční a koncovou hodinu odběrů pro daný den v týdnu. Tato doba je poté rozdělena na, již zmiňované, časové jednotky. Následně skript kontroluje, kolik dárců je již přihlášeno k odběru krve/plazmy v daný den a určitý čas (časovou jednotku). Pokud je počet nižší než maximální možný počet dárců na časovou jednotku, pak je tento blok zelený a umožňuje dárci, se na tento termín přihlásit. Pokud ovšem již počet přihlášených dárců dosáhl maxima, je tento čas zobrazen červeně a dárce si musí zvolit jiný čas odběru. Postup je znázorněn na vývojovém diagramu na Obrázek 15 Vývojový diagram tabulky časů.

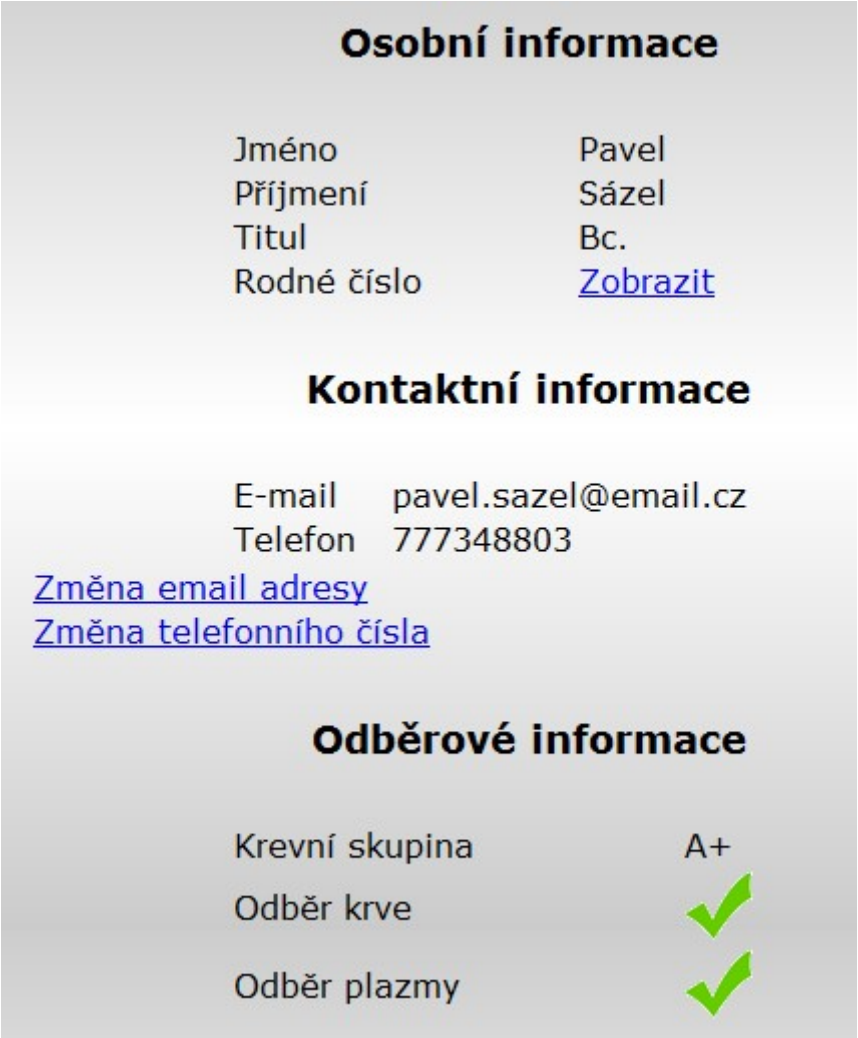
Obrázek 14 Výběr časů odběrů



Obrázek 15 Vývojový diagram tabulky časů

6.3 Zobrazení informací o dárci, editace kontaktních údajů

Tato stránka slouží dárci pro kontrolu správnosti údajů uložených v databázi, pro změnu kontaktních údajů. Výřez obrazovky je na Obrázek 16.



Obrázek 16 Obrazovka informací o dárci

Rodné číslo je při vypisování, jako citlivá informace, skryté. Po kliknutí na "Zobrazit", se vypíše. V případě, že by se dárci v menu nezobrazily možné odběry, pak si na této stránce může překontrolovat, zda má daný odběr povolen, či nikoliv. Na Obrázek 16 jsou v "Odběrových informacích" zobrazeny dvě zelené fajfky, které znamenají, že dárce má oba odběry povoleny. V opačném případě by fajfky byly nahrazeny červenými křížky.

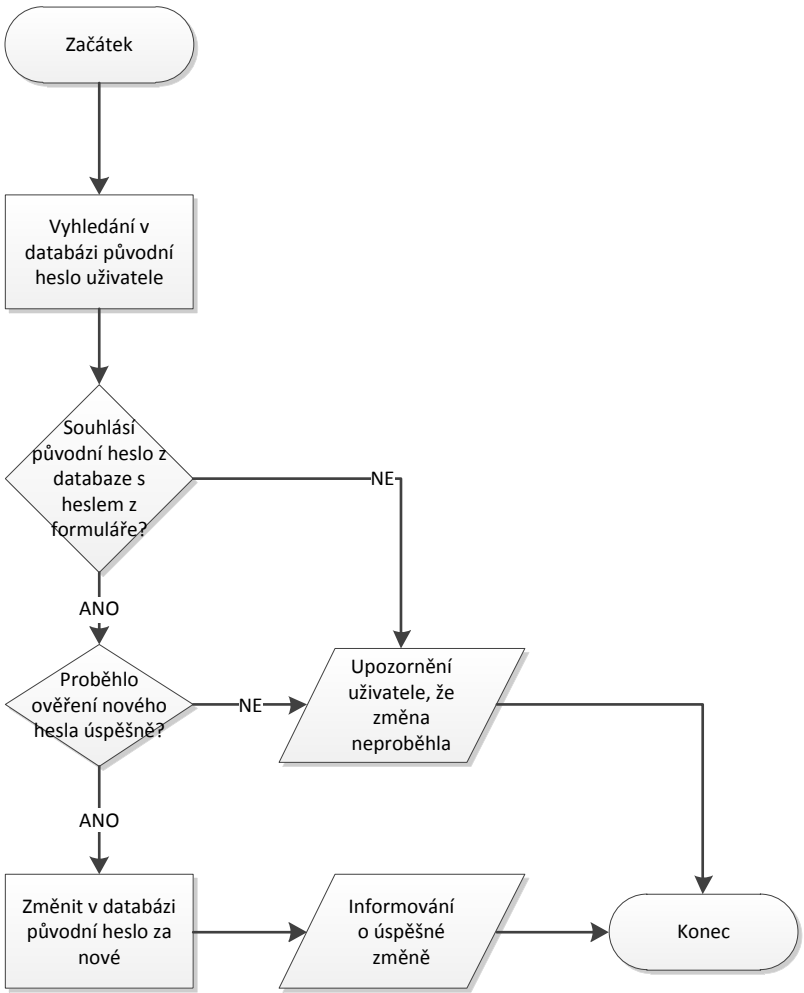
V případě založení nové emailové adresy nebo nového telefonního čísla, se na obrazovce zobrazí vstupní textové pole, do kterého dárce zapíše nové telefonní číslo, či emailovou adresu. Po stisku tlačítka "Potvrdit", pak je daný údaj v databázi aktualizován.

6.4 Změna hesla pro vstup do aplikace

Po registraci dárce do aplikace, je vygenerováno pět náhodných alfanumerických znaků, sloužící pro prvotní přihlášení dárce do aplikace. Pro snadnější zapamatování svého hesla, má dárce možnost si jej změnit. V aplikaci je použit formulář obsahující tři vstupní textové pole pro vložení původního hesla, nového hesla a potvrzení nového hesla. Maximální délka nového hesla je databází omezena na dvacet alfanumerických znaků. Délka nového hesla je

kontrolována funkcí napsanou v JavaScriptu. Pokud uživatel tuto délku překročí, je na to upozorněn a řetězec napsaný ve vstupním textovém poli je vymazán.

Samotný PHP skript na začátku podle ID čísla uživatele nalezne v databázi jeho aktuální heslo. Toto heslo je následně porovnáváno s řetězcem vepsaným do textového pole "Původní heslo". Pokud heslo souhlasí, probíhá ověření, zda řetězec "Nové heslo" souhlasí s heslem pro ověření. Pokud nějakou kontrolou řetězec neprojde, pak je uživatel informován, že heslo změněno nebylo. V opačném případě je původní heslo v databázi nahrazeno heslem novým. Vývojový diagram skriptu je na Obrázek 17.



Obrázek 17 Vývojový diagram změny hesla

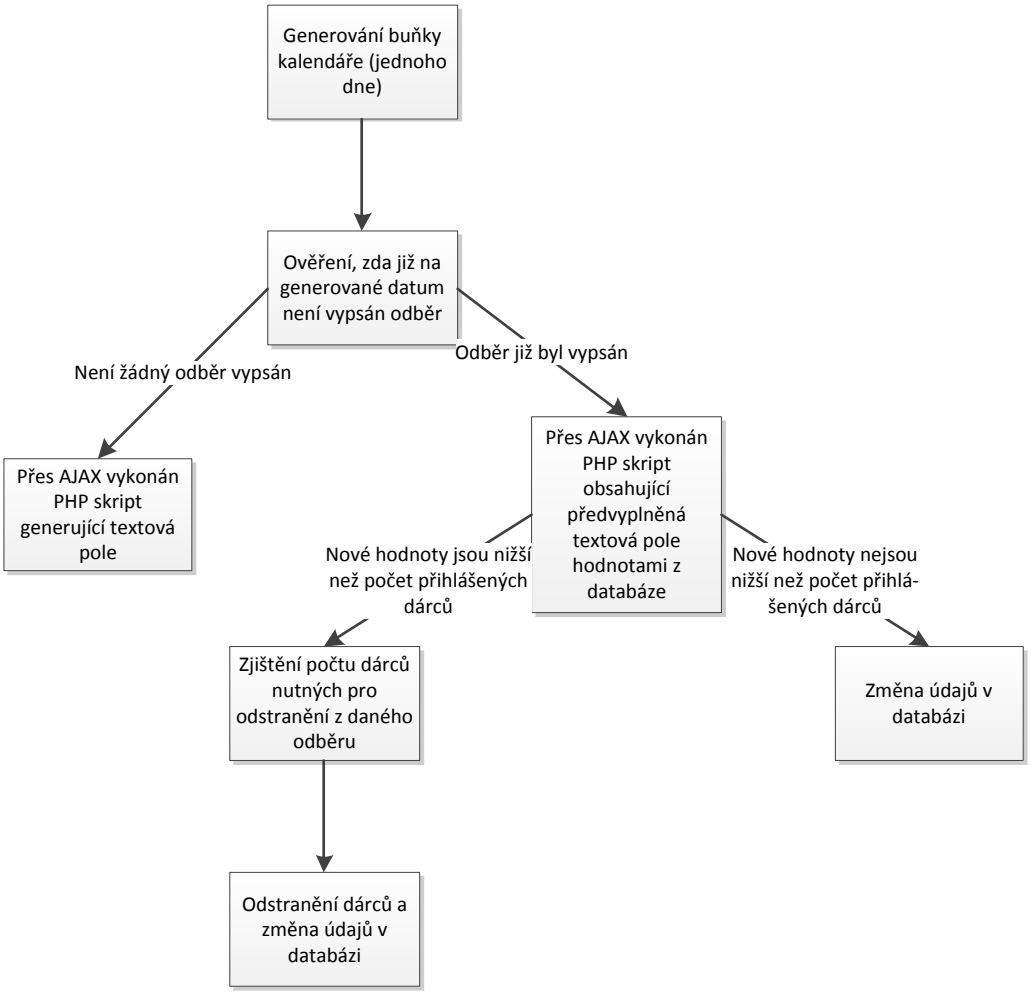
7 Organizační část

7.1 Vypsání a editace odběrů

Úlohou této části v celé aplikaci je zápis konkrétního typu odběru (krev nebo plazma) pro určité datum. Požadavek byl takový, aby pracovnice krevního centra, která uskutečňuje rozvrh odběrů, měla možnost vypsat pro konkrétní datum počet odběrů pro jednotlivé krevní skupiny.

V menu aplikace, má uživatel na výběr zapsání a editaci odběrů krve nebo plazmy. Zvolený typ odběru je přenesen v odkazu do skriptu generující kalendář uvedeném v 6.2.1. Místo kontroly generovaného data, kterou pokračuje generování kalendáře z odstavce 6.2.1 je v databázi kontrolováno, zda pro generovanou buňku kalendáře již existuje vypsaný termín odběru. Podle toho je vytvořen HTML tag odkazu s určitými parametry. Pokud v generovaném datu není žádný termín vypsán, je do parametru odkazu zapsána funkce JavaScriptu, používající AJAX, který odkazuje na PHP skript, jenž do stránky vypíše pro každou krevní skupinu jedno textové pole, do kterého uživatel zadá počet dárců, kteří se mohou na vypisované datum přihlásit.

Pokud datum již vypsáno je, je do HTML tagu generován opět JavaScript s AJAXem, který odkazuje na PHP skript vytvářející již předvyplněné textové pole. Po kliknutí na tlačítko "Uložit změny" je vykonán PHP skript, který zkontroluje, zda upravené počty odběrů nejsou menší, než je dosavadní počet přihlášených dárců na tyto upravované odběry. Pokud se naskytne situace, že počet dárců bude vyšší, pak je určitý počet přihlášených uživatelů s krevní skupinou, nebo krevními skupinami, kterých se snížení počtu odběrů týká, automaticky odhlášen a dárcům je zaslán informační email o zrušení jejich odběru.



Obrázek 18 Stručný popis postupu při zápisu nových odběrů nebo editace již zadaných odběrů

7.2 Zrušení všech odběrů v určitém datu

I zde má uživatel možnost zrušit odběry plazmy nebo krve zvlášť. Oproti výše uvedené editaci odběrů však odstraní celý řádek, který je vždy pro jedno určité datum, z databáze. To znamená, že zruší odběry pro všechny krevní skupiny. I v tomto případě jsou všichni dárci, kteří již byli na toto datum přihlášení, informováni o zrušení jejich odběru.

7.3 Správa uživatelů

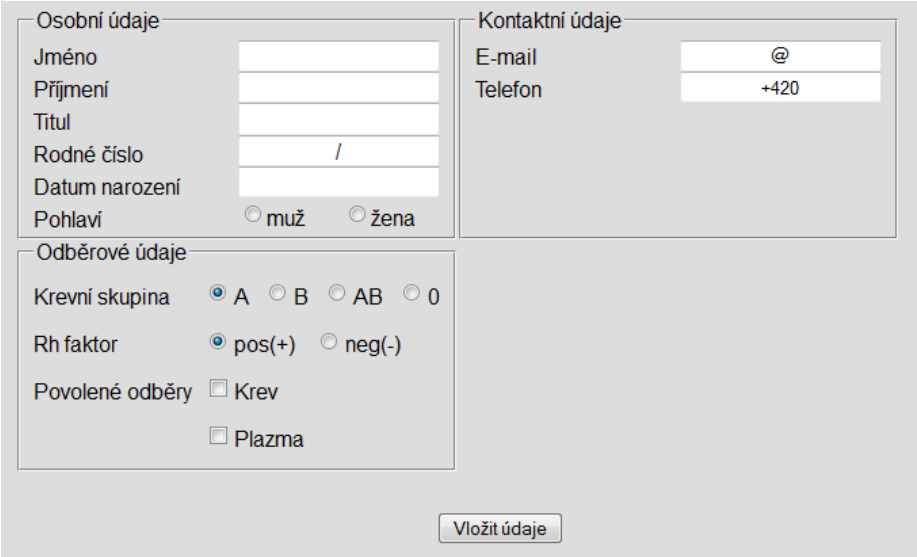
Přidat nové uživatele a spravovat existující mohou pouze uživatelé s právem "Administrátor" nebo "Organizátor". Odkazy pro jejich správu jsou v menu umístěny pod položkou "Dárci".

7.3.1 Vložení nového uživatele

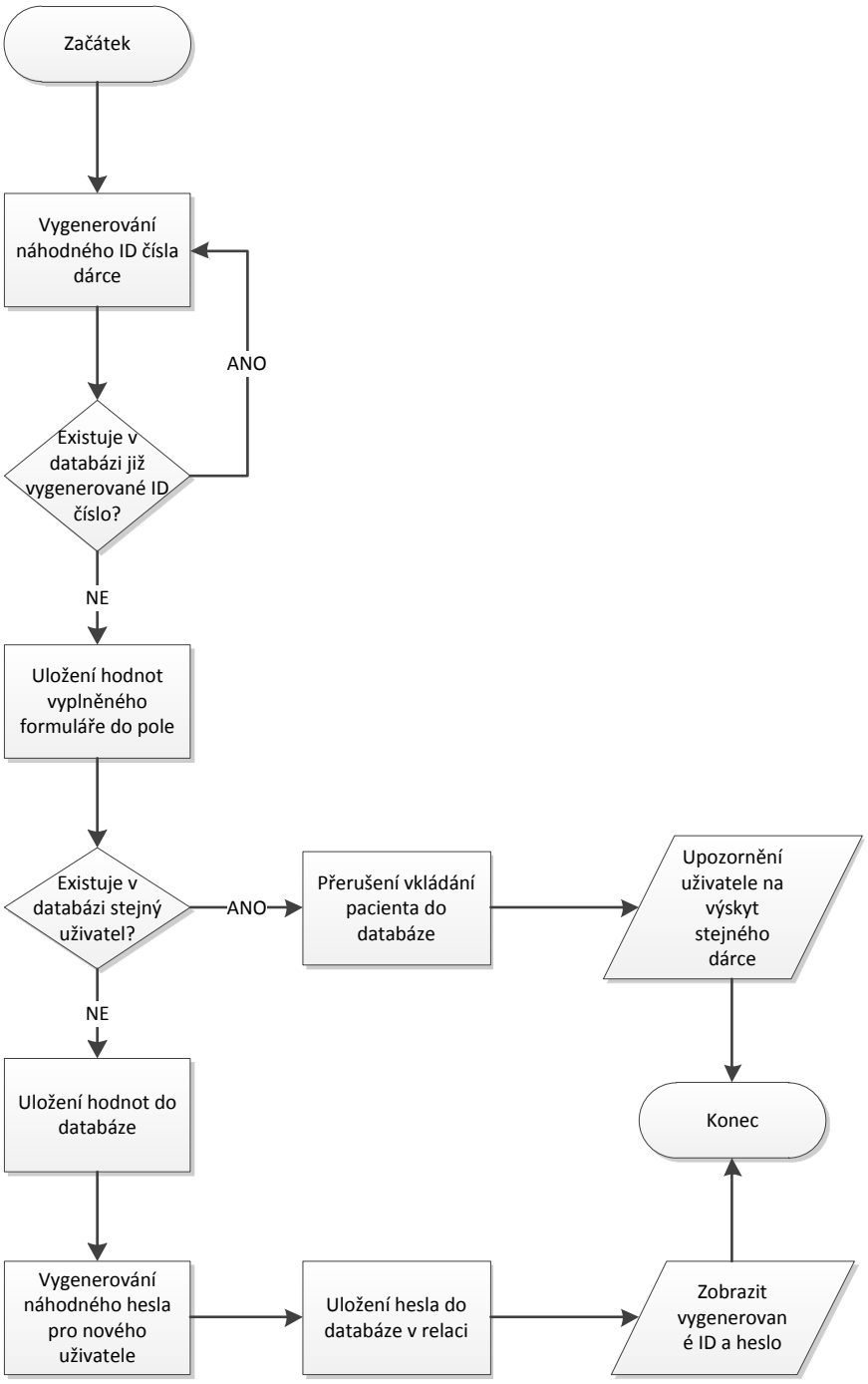
Pro usnadnění a zrychlení této akce by měla být vytvořená funkce importující seznam dárců z již zavedeného informačního systému krevního centra. Tato funkce není součástí této diplomové práce, více viz. kapitola 7.6.

Vkládání nových dárců do databáze je také realizováno postupným vkládáním. Formulář pro vložení nového dárce do databáze je na Obrázek 19. Zde je pro ověření vstupních hodnot textových polí použit JavaScript. Po zadání rodného čísla je automaticky doplněno datum narození dárce a jeho pohlaví. Obě tyto vstupní proměnné je možno dále editovat, a to z důvodu, že jejich odvozování z rodného čísla není vždy podle přesných pravidel a existují i výjimky, takže v případě doplnění nesprávných hodnot do automaticky vyplňovaných je lze upravit.

Dalšími kontrolovanými hodnotami je vždy po vypsání textového pole, formát vloženého řetězce (např. zda formát vloženého řetězce e-mail adresy souhlasí s obecným formátem tohoto řetězce).



Obrázek 19 Formulář pro vložení nového dárce



Obrázek 20 Vývojový diagram vložení nového dárce

Po odeslání vyplněného formuláře probíhá vygenerování náhodného sedmi místného identifikačního čísla. Toto číslo je pak následně ověřeno, zda se již v databázi nevyskytuje. Jelikož je sloupec IdUzivatele v tabulce uživatelů nastaven jako primární klíč, není možné mít dva účty se stejným číslem. Ověření čísla je funkčně primární kontrolou, a pokud se v databázi již vyskytuje stejné číslo, je vygenerováno nové identifikační číslo. Poté jsou hodnoty vyplněného formuláře uloženy do pole a je v databázi kontrolován výskyt stejného rodného čísla. Pokud se v databázi již nachází, je uživatel upozorněn, že tento dárce v databázi již existuje. V opačném případě jsou hodnoty z formuláře uloženy do tabulky "uzivatel" a následně

je vygenerováno heslo o délce pěti znaků a to je uloženo do tabulky "pass", která má relaci s tablukou "uzivatel". Nakonec je vypsáno ID uživatele a heslo na obrazovku.

7.3.2 Vyhledávání a editace uživatelů

Tato funkce slouží pro vyhledání jednoho konkrétního nebo více uživatelů a jejich následné editaci osobních údajů.
Vyhledává se podle těchto kritérií:

- Krevní skupiny
- Příjmení
- Rodného čísla
- ID dárce

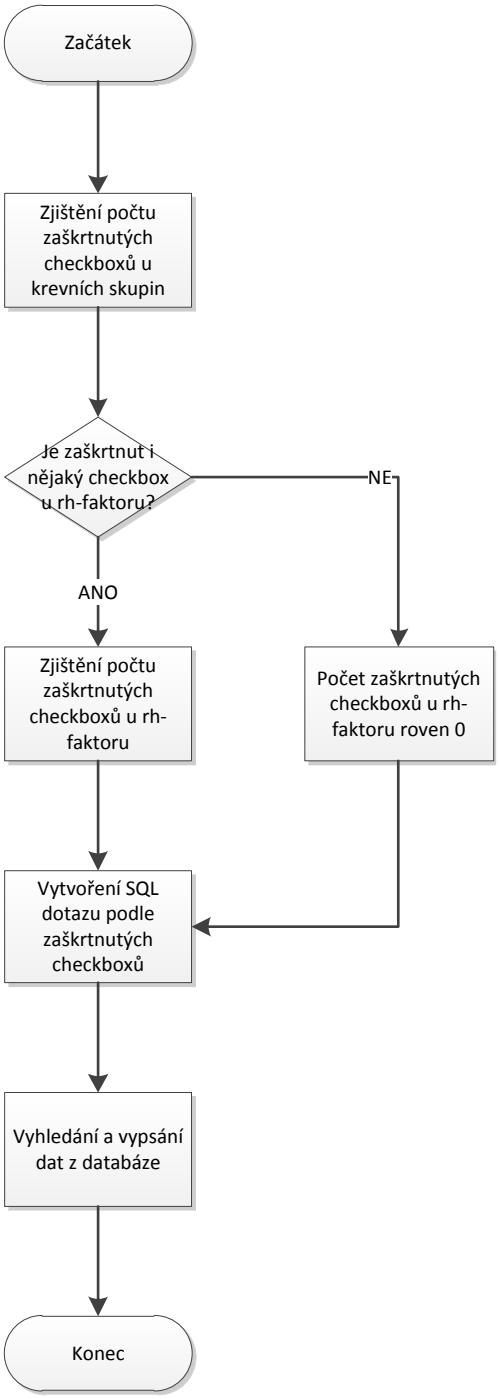
Obrázek 21 Ukázka formuláře pro vyhledávání podle krevní skupiny

Po stisknutí tlačítka "Vyhledat" je do tabulky pod formulář vypsán seznam uživatelů splňující kritéria vyhledávání (Obrázek 22).

ID dárce	Jméno	Příjmení	Titul	Rodné číslo	E-mail	Telefon	Krevní skupina	Rh faktor	Odběr krve	Odběr plazmy
0165801	Petr	Sázel		910410/1288	ronaldo91@seznam.cz	+420777485963	B	neg	A	A
1492807	Pavel	Sázel	Bc.	[illegible]25	pavel.sazel@email.cz	777348803	A	pos	A	A

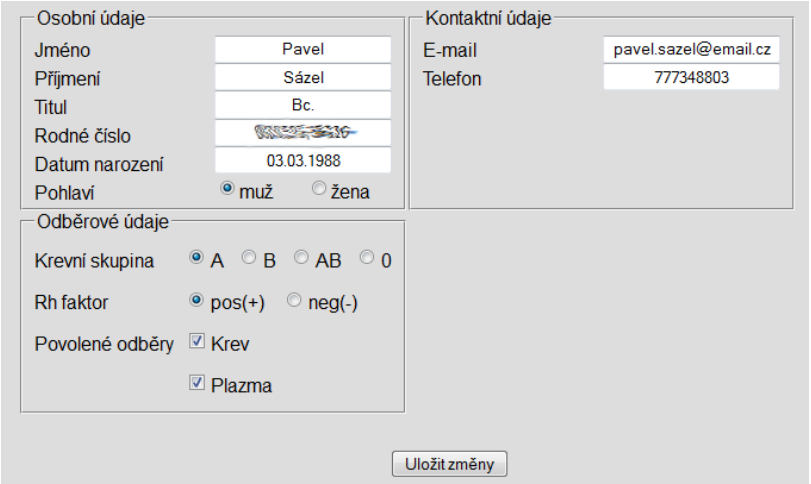
Obrázek 22 Výsledek vyhledávání

Při vyhledávání je možno vyhledávat podle všech kombinací označených zaškrtávacích políček ve formuláři. Od vypsání jedné přesně určené krevní skupiny s rh-faktorem, po vypsání všech dárců v databázi, tedy všech krevních skupin. Proces vyhledávání podle všech krevních skupin je znázorněn vývojovým diagramem na Obrázek 23.



Obrázek 23 Vývojový diagram vyhledávání podle krevních skupin

Ve výsledku vyhledávání je pak možnost údaje vyhledaného dárce upravit. To je zajištěno odkazem na skript, který s sebou nese i ID číslo dárce. Tento skript pak podle ID čísla vyhledá data dárce v databázi a vypíše je do stejného formuláře, který je použit pro vložení nového dárce (Obrázek 24). Po vykonání všech změn se pak data v databázi aktualizují.



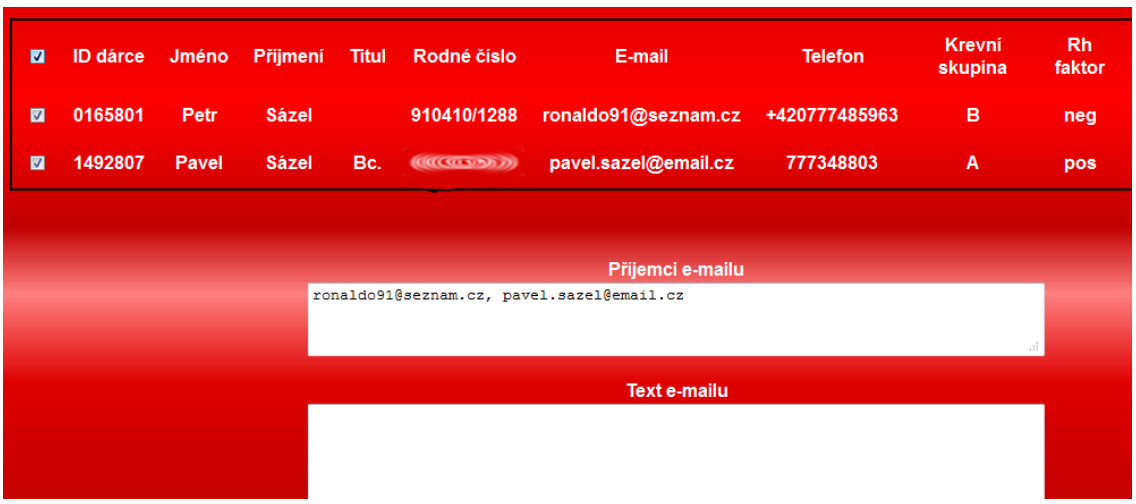
Obrázek 24 Formulář editace vyplněný daty z databáze

7.4 Odesílání elektronické pošty dárcům

Pro tuto funkci je použita funkce vyhledávání dárců z kapitoly 7.3.2. Může sloužit k zasílání elektronické pošty určitému dárci nebo větší skupině dárců. Oproti funkci vyhledávání a editace je ke každému výsledku vyhledávání připojeno zaškrtávací políčko. Email je pak zaslán označeným dárcům. Pokud by se naskytla potřeba odeslat e-mail dárcům jedné krevní skupiny, tak výsledků by bylo mnoho a označení každého zaškrtávacího políčka zvlášť by bylo velice nepraktické. Toto bylo vyřešeno funkcí JavaScriptu, která označí všechna tato políčka nalezených výsledků.

Emailové adresy označených dárců jsou vkládány do textového pole pod tabulkou výsledků. Slouží jako kontrola zvolených příjemců elektronické pošty.

Za tímto textovým polem následuje další pole, které je určeno již pro samotný obsah emailu.

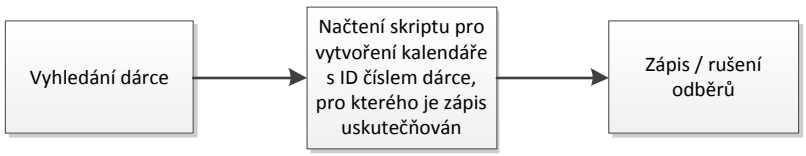


Obrázek 25 Odesílání emailu

Pro samotné odeslání emailu je použita knihovna PHPMailer[3]. Emaily lze posílat i přímo z PHP funkcí "mail()". S touto funkcí jsou trochu starosti, jelikož nesmí být opomenuto správné uvedení hlaviček a správné kódování. Knihovna PHPMailer zastřešuje vše okolo zasílání emailů jak přes funkci "mail()", tak i skrz existující SMTP server.

7.5 Přihlášení / odhlášení dárce k odběru

Tato funkce slouží pro případ, kdy by pracovnice krevního centra přihlašovala uživatele za něj. Pro tuto funkci je použita kombinace již výše zmiňovaných skriptů. Nejdříve je potřeba vybrat uživatele, který má být přihlášen. K tomu je využita funkce vyhledávání uvedená v kapitole 7.3.2. Vyhledávání je možno podle příjmení, rodného čísla nebo identifikačního čísla dárce. Výsledek je vypsán do tabulky. Identifikační číslo je generováno jako odkaz na skript pro generování kalendáře s odesílanou proměnnou právě identifikačního čísla zapisovaného dárce. Další postup je již stejný jako v kapitole 6.2. Ve vygenerovaném kalendáři se uživatel s právem "Organizator" nebo "Admin" pohybuje jakoby byl zapisovaný uživatel. Vidí všechny jeho zapsané odběry, má možnost rušit odběry za dárce nebo jej zapisovat na vypsané a volné termíny odběrů.



Obrázek 26 Postup pro editaci odběrů zvoleného dárce

7.6 Import / Export dat

7.6.1 Import dat

Import dat při řešení této práce zůstal nedořešeným tématem. Navrhován byl import přímo databázových exportovaných tabulek MySQL popř. soubor XML. Jelikož formát dat pro importování musí korelovat s možnostmi formátu pro import aplikace. V období psaní a řešení této práce, probíhala v krevním centru, se kterým byla tato záležitost konzultována, změna informačního systému a nebyl znám formát exportovaných dat z tohoto systému.

7.6.2 Export dat

Export dat je zaměřen na výpis přihlášených uživatelů pro určitý den, nebo pro rozmezí dnů. Výstupní soubor je formátu PDF, jehož ukázka je na Obrázek 29. Generování souboru PDF je realizováno pomocí knihovny FPDF[4]. Tato knihovna je volně dostupná na internetu jako OpenSource projekt a je k dispozici jako freeware jak pro soukromé, tak i pro komerční účely.

[3] Oficiální stránky knihovny PHPMailer - http://phpmailer.worxware.com [28.4.2012]

[4] Oficiální stránky knihovny FPDF - www.fpdf.org [28.4.2012]

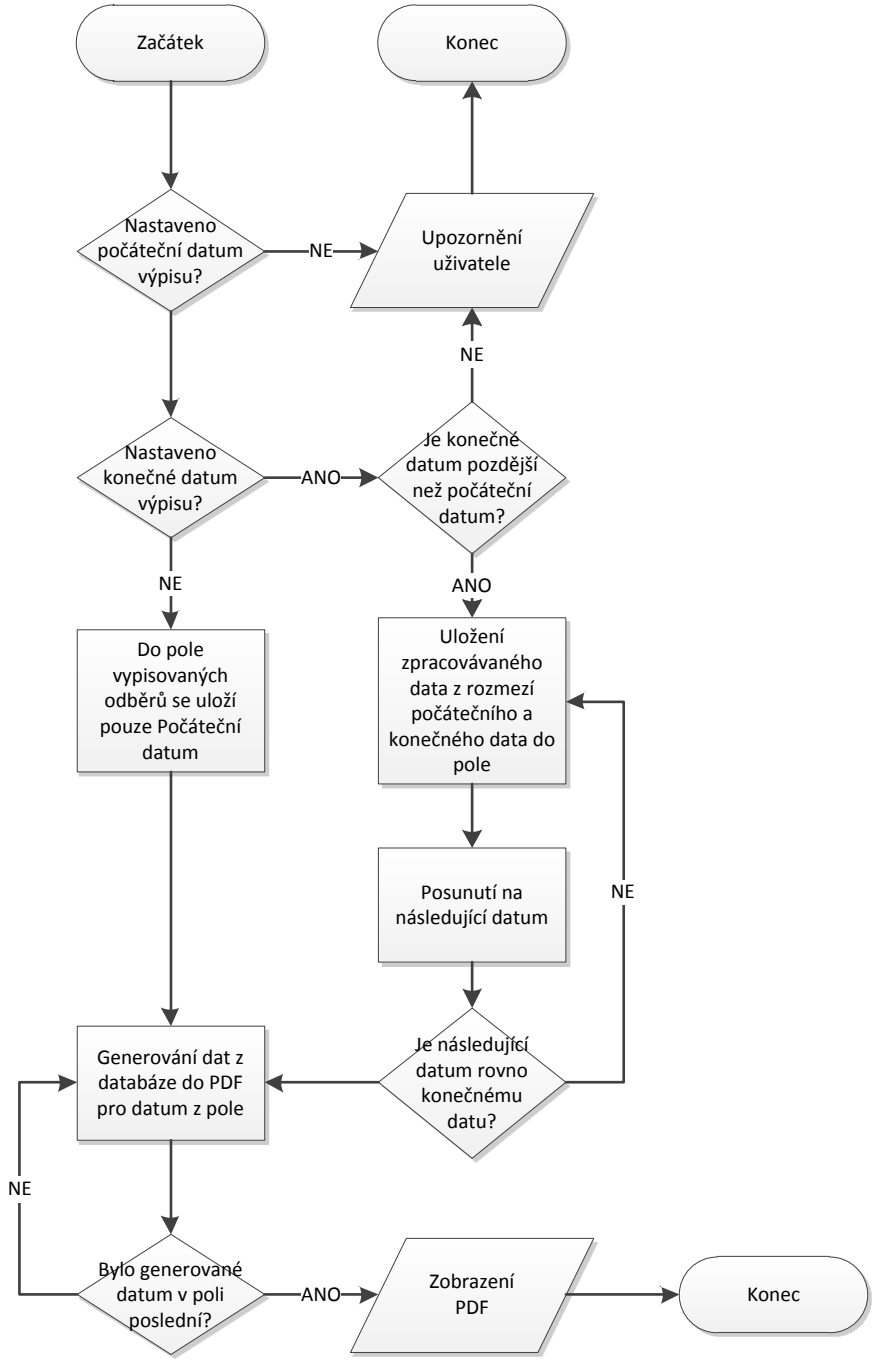
Nevýhodou této knihovny však je to, že obsahuje pouze typy fontů bez českých znaků. Tento problém je řešen přes, na internetu dostupný, online konvertor[5], který zajistí překonvertování existující sady fontů, např. z MS Windows, do souboru fontů pro knihovnu FPDF. Výstupní soubory z tohoto konvertoru se pak vloží do adresáře "font", umístěném v adresáři FPDF knihovny. Při tvorbě samotného PDF se poté jen tento nový font přidá.

Další problém nastal s kódováním řetězců. V aplikaci je použito kódování UTF-8, které ale tato knihovna nepodporuje. Řešením je jednoduchý PHP příkaz ICONV(), kterým se kódování řetězce převede z UTF-8 například na Windows-1250, které již knihovna podporuje.

Samotný formulář exportu dat je znázorněn na Obrázek 28. Pro rychlejší vyplnění položek a ve správném formátu požadovaném aplikací, je ještě použit kalendář napsaný JavaScriptem. Ten se objeví po kliknutí na ikonku v textboxu pro zadání data.

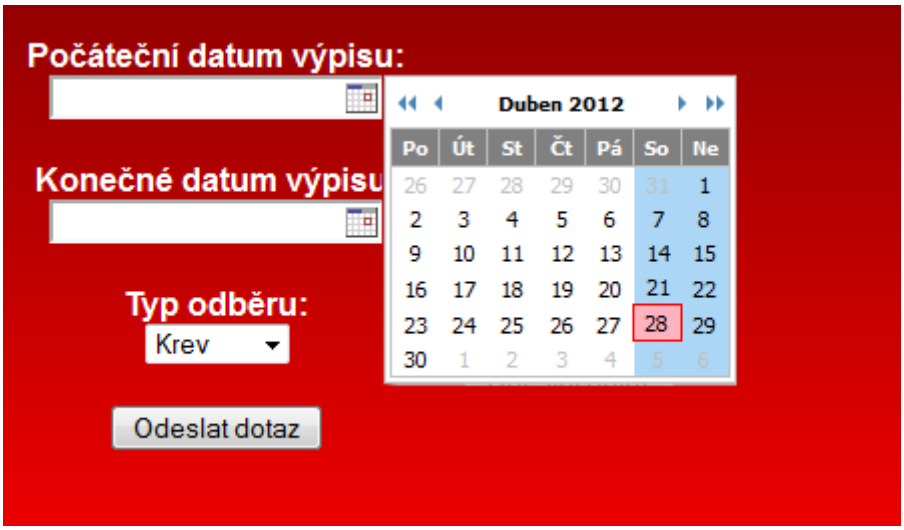
Položka "Počáteční datum výpisu" je povinná, a pokud není vyplněná, je skript ukončen ještě před jakýmkoliv generováním dat do souboru PDF. Další kontrolou je, že počáteční datum výpisu musí být menší než konečné datum výpisu. Celý proces je znázorněn na vývojovém diagramu na Obrázek 27.

[5] Oficiální stránky konvertoru fontů - fpdf.fruit-lab.de [28.4.2012]



Obrázek 27 Vývojový diagram skriptu pro export přihlášených uživatelů do PDF

Pokud uživatel vyplní pouze položku "Počáteční datum výpisu" - je vygenerováno PDF, které obsahuje seznam přihlášených uživatelů pouze na datum, které uživatel zadal. Pokud jsou zadána obě data, je pro každé datum v dokumentu PDF vytvořena nová stránka, aby celková struktura dokumentu byla přehledná. Pokud se objeví případ, že by na generované datum nebyl vypsán termín odběrů nebo na toto datum nebyl přihlášen žádný uživatel, pak je i pro toto datum vytvořena stránka obsahující prázdnou tabulku, a to také z důvodu přehlednosti a omezení pochybnosti, že by se seznam přihlášených uživatelů vygeneroval nesprávně.



Obrázek 28 Formulář exportu dat

Výpis naplánovaných odběrů

Datum: 03/04/2012

Typ odběru: Krev

Datum	Čas	ID uživatele	Jméno	Příjmení	Krevní sk.
03/04/2012	10:00	1492807	Pavel	Sázel	A pos
03/04/2012	11:00	0165801	Petr	Sázel	B neg

Obrázek 29 Výsek exportovaného PDF

8 Administrátorská část

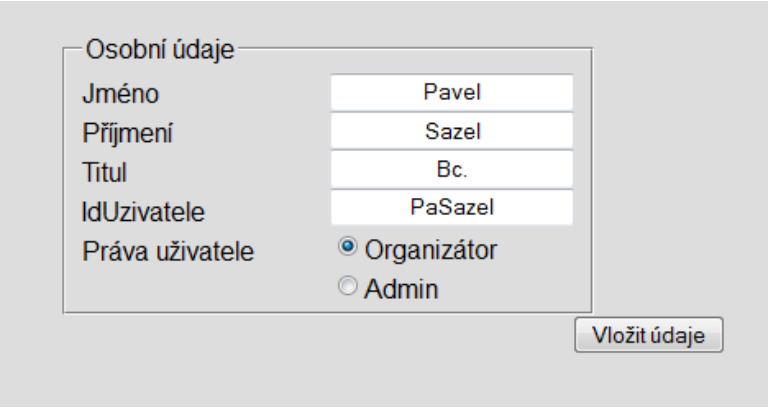
Administrátorská část je část aplikace, která má plný přístup ke všem jejím funkcím. Obsahuje všechny funkce organizační části uvedené v kapitole 7.

Obrázek 30 Menu funkcí administrátorského účtu

Umožňuje tedy vypisovat odběr pro konkrétní dny, již zapsané odběry editovat nebo je zcela zrušit.

8.1 Vkládání a editace uživatelů s rozšířenými právy

Kromě vložení nového dárce, má již stávající uživatel s právem "Admin" možnost vložit i nového uživatele rovněž s právem "Admin" nebo "Organizátor". Informace o stávajícím uživateli může editovat a měnit práva přístupu či jej z databáze uživatelů s rozšířenými právy odstranit. Předpokládá se, že administrátorských účtu bude minimální počet a tito uživatelé budou spolehlivé osoby, jelikož mohou v plném rozmezí nakládat s účty všech uživatelů.



Obrázek 31 Formulář pro vložení a editaci uživatelů s rozšířenými právy

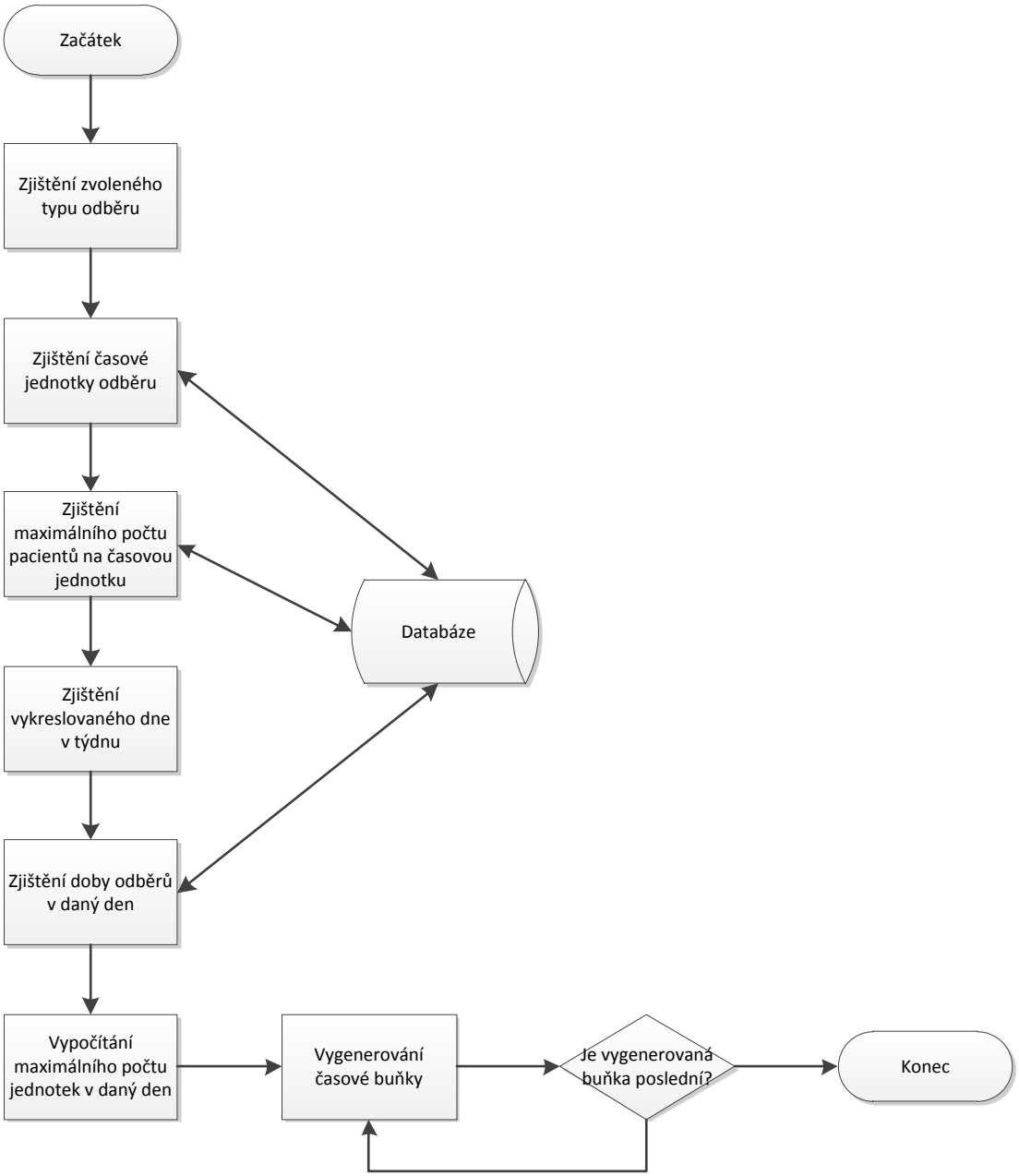
Při vkládání nového uživatele s rozšířenými právy do databáze si může uživatel zvolit jakékoliv ID, avšak maximálně v počtu deseti znaků.

Po zadání jména a příjmení je toto ID funkcí Javascriptu vygenerováno automaticky jako spojení prvních dvou písmen ze jména a dalších osm znaků z příjmení. Pokud příjmení uživatele má méně než osm znaků, pak je ID vytvořeno z celého příjmení. Toto ID je zbaveno

interpunkce, aby se omezily chybně zadané vstupní informace při přihlašování. Toto automaticky vygenerované ID není nutno použít. Po jeho vygenerování je možno jej změnit dle iniciativy uživatele.

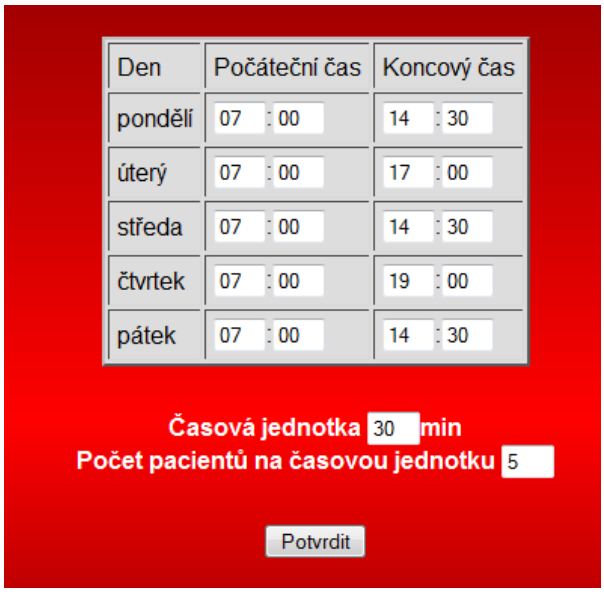
8.2 Administrace odběrových údajů

Další funkci, kterou administrátorská část obsahuje, je změna časů odběrů v jednotlivých dnech. Lze tedy pro každý den a pro každý typ odběru nastavit počáteční čas a konečný čas odběrů. Krom toho se zde definuje také časová jednotka pro typ odběru. Čas odběru krve se pohybuje v průměru mezi 8-12 minutami a čas odběru plazmy mezi 30-120 minutami.



Obrázek 32 Použití nastavení pro generování časové tabulky

Délka časové jednotky tedy bude muset být zvolena co nejefektivněji, aby nevznikaly opožděné odběry dárců a zároveň, aby tato jednotka nebyla moc dlouhá a čas odběru kratší, což by způsobilo nechtěné volné místo.



Obrázek 33 Výsek obrazovky pro zadání časů odběrů plazmy

Na Obrázek 34 je uvedena vygenerovaná tabulka pro den, kde je začátek odběrů v 7:00 a konec v 14:30. Časová jednotka v tomto případě byla 30minut. Dále skript kontroluje maximální možný počet dárců na tuto časovou jednotku. V případě, kdy by tento počet byl roven například pěti a právě tento počet by již byl přihlášen na čas 8:00, pak by tato buňka byla červená a další dárci by se na tento čas již přihlásit nemohli. Poslední vygenerovaná časová buňka je v dostatečném rozmezí ke konečnému času odběrů tak, aby dárci přihlášení na tento čas nepřekročili konečnou dobu odběrů.

Obrázek 34 Generovaná časová tabulka

9 Závěr

Výsledkem této diplomové práce je internetová aplikace, která by se do praktického využití měla dostat během pár týdnů, a která by měla být nasazena pro efektivní rozvrh odběrů krve a plazmy, a pohodlné přihlašování dárců na ně. Krevnímu centru by aplikace měla časově ulehčit od komunikace s dárci ohledně přihlašování a poskytnout tak prostor pro jiné využití zaměstnanců krevního centra, jelikož počet dárců navštěvující Krevní centrum Fakultní nemocnice Ostrava se pohybuje kolem 10,5 tisíc.

Celá aplikace byla naprogramována ve třech programovacích jazycích - HTML, PHP a JavaScript. Jelikož na použité vývojové prostředky nebyly kladeny žádné specifikace, tak byly zvoleny prostředky, které dle mého názoru jsou dostatečně silné a bezpečné pro aplikaci jako je rozvrhovací systém krevního centra a zároveň je jejich cena minimální, což je s ohledem na využití v praxi velmi podstatná věc.

Vývoj aplikace proběhl v operačním systému Windows 7 a funkci webového serveru zajistil program Complex Web Server. Následné testování bylo uskutečněno v prohlížečích Internet Explorer 6, 7 a 9, Mozilla Firefox 9.0.1, Apple Safari 5.1.5 pro OS Windows 7 a Opera 11.62 - tedy velká část používaných prohlížečů.

Poslední dobou se pro psaní webových stránek rozšířilo používání jazyka XHTML (eXtensible HyperText Markup Language) neboli rozšířitelný hypertextový značkovací jazyk, který se vyvinul z jazyka HTML jako XML aplikace. Pro tuto práci však byl zvolen jazyk HTML (HyperText Markup Language), protože dle mého názoru je jazyk XHTML slepou vývojovou větví.

Jelikož si myslím, že tato aplikace má jistou budoucnost v praktickém využití, rád bych na ní, společně s krevním centrem Fakultní nemocnice Ostrava, pracoval i po jejím odevzdání jakož to diplomové práce.

Jedním z požadavků krevního centra bylo, aby si dárce mohl zvolit odběr plazmy v pěti nebo sedmi cyklech odběrů, které následují po sobě. Přímo tato možnost, aby dárci byly navrhnuty termíny všech sedmi cyklů a dárce si je popřípadě mohl změnit, aplikace neobsahuje. Toto je zatím řešeno způsobem, kdy si dárce v kalendáři těchto sedm odběrů naplánuje zvlášť.

Možných funkcí a možností, kterými by se tato aplikaci mohla dále vylepšit je mnoho a já budu moc rád, když se mi časem podaří ji dovést do takového stavu, kdy počet požadovaných inovací na ní bude minimální.

10 Použitá literatura

[1] Michael PEACOCK, *PHP 5 E-Commerce Development*, Computer Press, Brno, 2011, ISBN 978-80-251-3181-7

[2] Kevin YANK, Cameron ADAMS, *Simply JavaScript*, Zoner Press, Brno, 2008, ISBN 978-80-86815-94-7

[3] Petr STANÍČEK, *CSS Kaskádové styly - Kompletní průvodce*, Computer Press, Praha, 2003, ISBN 80-7226-872-4

[4] Jiří KOSEK, *PHP - Tvorba interaktivních internetových aplikací,* Grada Publishing, Praha, 1998, ISBN 80-7169-373-1

[5] Jesus CASTAGNETTO, Harish RAWAT, Sascha SCHUMANN, Chris SCOLLO, Deepak VELIATH, *PHP programujeme profesionálně*, Computer Press, Praha, 2001, ISBN 80-7226-310-2

[6] Bruce SCHNEIER, *Applied Cryptography: protocols, algorithms, and source code in C*,

[7] HTML [http://php.vrana.cz/obrana-proti-sql-injection.php , 21.4.2012]

[8] HTML [http://cs.wikipedia.org/wiki/Autentizace , 21.4.2012]

[9] HTML [http://php.vrana.cz/zabezpeceni-session-promennych.php , 21.4.2012]

[10] Steve SUEHRING, *JavaScript Krok za krokem*, Computer Press, Praha, 2008, ISBN 978-80-251-2241-9

[11] HTML [http://cs.wikipedia.org/wiki/HyperText_Markup_Language, 21.4.2012]

[12] HTML [http://php.vrana.cz/rychlost-vkladani-do-innodb-tabulek.php, 3.5.2012]

[13] HTML [http://ponkrac.net/complex-web-server/cs, 3.5.2012]

[14] HTML [http://cs.wikipedia.org/wiki/Apache_HTTP_Server, 3.5.2012]

[15] HTML [http://cs.wikipedia.org/wiki/Mysql, 3.5.2012]